DonorsChoose

DonorsChoose.org receives hundreds of thousands of project proposals each year for classroom projects in need of funding. Right now, a large number of volunteers is needed to manually screen each submission before it's approved to be posted on the DonorsChoose.org website.

Next year, DonorsChoose.org expects to receive close to 500,000 project proposals. As a result, there are three main problems they need to solve:

- How to scale current manual processes and resources to screen 500,000 projects so that they can be posted as quickly and as efficiently as possible
- How to increase the consistency of project vetting across different volunteers to improve the experience for teachers
- How to focus volunteer time on the applications that need the most assistance

The goal of the competition is to predict whether or not a DonorsChoose.org project proposal submitted by a teacher will be approved, using the text of project descriptions as well as additional metadata about the project, teacher, and school. DonorsChoose.org can then use this information to identify projects most likely to need further review before approval.

About the DonorsChoose Data Set

The `train.csv` data set provided by DonorsChoose contains the following features:

Feature	Description
`project_id`	A unique identifier for the proposed project. **Example:** `p0`
`project_title`	Title of the project. **Examples:** • `Art Will Make You Happy!` • `First Grade Fun`
`project_grade_category`	Grade level of students for which the project is targeted. C enumerated values: • `Grades PreK-2` • `Grades 3-5` • `Grades 6-8` • `Grades 9-12`
`project_subject_categories`	One or more (comma-separated) subject categories for th following enumerated list of values: • `Applied Learning` • `Care & Hunger` • `Health & Sports` • `History & Civics` • `Literacy & Language` • `Math & Science` • `Music & The Arts` • `Special Needs` • `Warmth` **Examples:** • `Music & The Arts` • `Literacy & Language, Math & Science`
`school_state`	State where school is located (Two-letter U.S. postal code (https://en.wikipedia.org/wiki/List_of_U.S._state_abbrevia **Example:** `WY`
`project_subject_subcategories`	One or more (comma-separated) subject subcategories fo **Examples:** • `Literacy` • `Literature & Writing, Social Sciences`

Feature	Description
project_resource_summary	An explanation of the resources needed for the project. **E** • My students need hands on literacy materia sensory needs!
project_essay_1	First application essay*
project_essay_2	Second application essay*
project_essay_3	Third application essay*
project_essay_4	Fourth application essay*
project_submitted_datetime	Datetime when project application was submitted. **Examp** 12:43:56.245
teacher_id	A unique identifier for the teacher of the proposed project bdf8baa8fedef6bfeec7ae4ff1c15c56
teacher_prefix	Teacher's title. One of the following enumerated values: • nan • Dr. • Mr. • Mrs. • Ms. • Teacher.
teacher_number_of_previously_posted_projects	Number of project applications previously submitted by th **Example:** 2

* See the section **Notes on the Essay Data** for more details about these features.

Additionally, the `resources.csv` data set provides more data about the resources required for each project. Each line in this file represents a resource required by a project:

Feature	Description
id	A `project_id` value from the `train.csv` file. **Example:** `p036502`
description	Desciption of the resource. **Example:** `Tenor Saxophone Reeds, Box of 25`
quantity	Quantity of the resource required. **Example:** `3`
price	Price of the resource required. **Example:** `9.95`

Note: Many projects require multiple resources. The `id` value corresponds to a `project_id` in train.csv, so you use it as a key to retrieve all resources needed for a project:

The data set contains the following label (the value you will attempt to predict):

Label	Description

Label	Description
project_is_approved	A binary flag indicating whether DonorsChoose approved the project. A value of 0 indicates the project was not approved, and a value of 1 indicates the project was approved.

Notes on the Essay Data

Prior to May 17, 2016, the prompts for the essays were as follows:

- __project_essay_1:__ "Introduce us to your classroom"
- __project_essay_2:__ "Tell us more about your students"
- __project_essay_3:__ "Describe how your students will use the materials you're requesting"
- __project_essay_3:__ "Close by sharing why your project will make a difference"

Starting on May 17, 2016, the number of essays was reduced from 4 to 2, and the prompts for the first 2 essays were changed to the following:

- __project_essay_1:__ "Describe your students: What makes your students special? Specific details about their background, your neighborhood, and your school are all helpful."
- __project_essay_2:__ "About your project: How will these materials make a difference in your students' learning and improve their school lives?"

For all projects with project_submitted_datetime of 2016-05-17 and later, the values of project_essay_3 and project_essay_4 will be NaN.

In [1]:
```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")

import sqlite3
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer

import re
# Tutorial about Python regular expressions: https://pymotw.com/2/re/
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

from gensim.models import Word2Vec
from gensim.models import KeyedVectors
import pickle

from tqdm import tqdm
import os

from plotly import plotly
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
from collections import Counter
```

```
C:\Users\samar\Anaconda3\lib\site-packages\gensim\utils.py:1197: UserWarning:
detected Windows; aliasing chunkize to chunkize_serial
  warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

1.1 Reading Data

In [2]:
```
project_data = pd.read_csv('train_data.csv')
resource_data = pd.read_csv('resources.csv')
```

In [3]:
```
print("Number of data points in train data", project_data.shape)
print('-'*50)
print("The attributes of data :", project_data.columns.values)
```

```
Number of data points in train data (109248, 17)
--------------------------------------------------
The attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'teacher_prefix' 'sc
hool_state'
 'project_submitted_datetime' 'project_grade_category'
 'project_subject_categories' 'project_subject_subcategories'
 'project_title' 'project_essay_1' 'project_essay_2' 'project_essay_3'
 'project_essay_4' 'project_resource_summary'
 'teacher_number_of_previously_posted_projects' 'project_is_approved']
```

In [4]:
```
print("Number of data points in train data", resource_data.shape)
print(resource_data.columns.values)
resource_data.head(2)
```

```
Number of data points in train data (1541272, 4)
['id' 'description' 'quantity' 'price']
```

Out[4]:

	id	description	quantity	price
0	p233245	LC652 - Lakeshore Double-Space Mobile Drying Rack	1	149.00
1	p069063	Bouncy Bands for Desks (Blue support pipes)	3	14.95

1.2 preprocessing of `project_subject_categories`

In [5]:
```
catogories = list(project_data['project_subject_categories'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python
cat_list = []
for i in catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp+=j.strip()+" " #" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_') # we are replacing the & value into
    cat_list.append(temp.strip())

project_data['clean_categories'] = cat_list
project_data.drop(['project_subject_categories'], axis=1, inplace=True)

from collections import Counter
my_counter = Counter()
for word in project_data['clean_categories'].values:
    my_counter.update(word.split())

cat_dict = dict(my_counter)
sorted_cat_dict = dict(sorted(cat_dict.items(), key=lambda kv: kv[1]))
```

In [6]:
```
##print(cat_list)
#cat_series=pd.Series(cat_list)
##print("cat_series",cat_series)
#cat_ser#ies.value_counts()
##cat_list_df=pd.DataFrame({'clean_cat':cat_list})
```

In [7]:
```
#cat_list_df_unique=cat_list_df.drop_duplicates()
#print(cat_list_df_unique)
```

In [8]:
```
#cat_list_df.
```

In [9]:
```
#print(project_data['clean_categories'])
```

In [10]:
```
#print(sorted_cat_dict)
```

1.3 preprocessing of `project_subject_subcategories`

In [11]:
```
sub_catogories = list(project_data['project_subject_subcategories'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

sub_cat_list = []
for i in sub_catogories:
    temp = ""
    # consider we have text like this "Math & Science, Warmth, Care & Hunger"
    for j in i.split(','): # it will split it in three parts ["Math & Scienc
e", "Warmth", "Care & Hunger"]
        if 'The' in j.split(): # this will split each of the catogory based on
space "Math & Science"=> "Math","&", "Science"
            j=j.replace('The','') # if we have the words "The" we are going to
replace it with ''(i.e removing 'The')
        j = j.replace(' ','') # we are placeing all the ' '(space) with ''(emp
ty) ex:"Math & Science"=>"Math&Science"
        temp +=j.strip()+" "#" abc ".strip() will return "abc", remove the tra
iling spaces
        temp = temp.replace('&','_')
    sub_cat_list.append(temp.strip())

project_data['clean_subcategories'] = sub_cat_list
project_data.drop(['project_subject_subcategories'], axis=1, inplace=True)

# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
my_counter = Counter()
for word in project_data['clean_subcategories'].values:
    my_counter.update(word.split())

sub_cat_dict = dict(my_counter)
sorted_sub_cat_dict = dict(sorted(sub_cat_dict.items(), key=lambda kv: kv[1]))
```

1.4 preprocessing of project_grade_category

In [12]:
```
prj_grade_cat = list(project_data['project_grade_category'].values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

prj_grade_cat_list = []
for i in prj_grade_cat:
    for j in i.split(' '): # it will split by space
        j=j.replace('Grades','') # if we have the words "Grades" we are going
 to replace it with ''(i.e removing 'Grades')
    prj_grade_cat_list.append(j.strip())

project_data['clean_grade'] = prj_grade_cat_list
project_data.drop(['project_grade_category'], axis=1, inplace=True)

# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
my_counter = Counter()
for word in project_data['clean_grade'].values:
    my_counter.update(word.split())

prj_grade_cat_dict = dict(my_counter)
sorted_prj_grade_cat_dict = dict(sorted(prj_grade_cat_dict.items(), key=lambda
kv: kv[1]))

project_data['clean_grade'].values
```

Out[12]:
```
array(['PreK-2', '6-8', '6-8', ..., 'PreK-2', '3-5', '6-8'], dtype=object)
```

1.5 preprocessing of teacher_prefix

In [13]:
```
#tea_pfx_cat = list(project_data['teacher_prefix'].values)
tea_pfx_cat = list(project_data['teacher_prefix'].astype(str).values)
# remove special characters from list of strings python: https://stackoverflo
w.com/a/47301924/4084039

# https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
# https://stackoverflow.com/questions/23669024/how-to-strip-a-specific-word-fr
om-a-string
# https://stackoverflow.com/questions/8270092/remove-all-whitespace-in-a-strin
g-in-python

##https://stackoverflow.com/questions/52736900/how-to-solve-the-attribute-erro
r-float-object-has-no-attribute-split-in-pyth
#vectorizer.fit(project_data['teacher_prefix'].astype(str).values)

tea_pfx_cat_list = []
for i in tea_pfx_cat:
    #for j in i.split(' '): # it will split by space
    #j=j.replace('.','') # if we have the words "Grades" we are going to repla
ce it with ''(i.e removing 'Grades')
    i=i.replace('.','') # if we have the words "Grades" we are going to replac
e it with ''(i.e removing 'Grades')
    i=i.replace('nan','') # if we have the words "Grades" we are going to repl
ace it with ''(i.e removing 'Grades')
    tea_pfx_cat_list.append(i.strip())

project_data['clean_tea_pfx'] = tea_pfx_cat_list
project_data.drop(['teacher_prefix'], axis=1, inplace=True)

# count of all the words in corpus python: https://stackoverflow.com/a/2289859
5/4084039
my_counter = Counter()
for word in project_data['clean_tea_pfx'].values:
    my_counter.update(word.split())

tea_pfx_cat_dict = dict(my_counter)
sorted_tea_pfx_cat_dict = dict(sorted(tea_pfx_cat_dict.items(), key=lambda kv:
kv[1]))

project_data['clean_tea_pfx'].values
```

Out[13]:
```
array(['Mrs', 'Mr', 'Ms', ..., 'Mrs', 'Mrs', 'Ms'], dtype=object)
```

1.6 Text preprocessing

In [14]:
```
# merge two column text dataframe:
project_data["essay"] = project_data["project_essay_1"].map(str) +\
                        project_data["project_essay_2"].map(str) + \
                        project_data["project_essay_3"].map(str) + \
                        project_data["project_essay_4"].map(str)
```

In [15]:
```
project_data.head(2)
```

Out[15]:

	Unnamed: 0	id	teacher_id	school_state	project_submi
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	IN	2016-12-05 13:
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	FL	2016-10-25 09:

Using Pretrained Models: TFIDF weighted W2V

In [16]:
```
# printing some random reviews
print(project_data['essay'].values[0])
print("="*50)
print(project_data['essay'].values[150])
print("="*50)
print(project_data['essay'].values[1000])
print("="*50)
print(project_data['essay'].values[20000])
print("="*50)
print(project_data['essay'].values[99999])
print("="*50)
```

My students are English learners that are working on English as their second or third languages. We are a melting pot of refugees, immigrants, and native-born Americans bringing the gift of language to our school. \r\n\r\n We have over 24 languages represented in our English Learner program with students at every level of mastery. We also have over 40 countries represented with the families within our school. Each student brings a wealth of knowledge and experiences to us that open our eyes to new cultures, beliefs, and respect.\"The limits of your language are the limits of your world.\"-Ludwig Wittgenstein Our English learner's have a strong support system at home that begs for more resources. Many times our parents are learning to read and speak English along side of their children. Sometimes this creates barriers for parents to be able to help their child learn phonetics, letter recognition, and other reading skills.\r\n\r\nBy providing these dvd's and players, students are able to continue their mastery of the English language even if no one at home is able to assist. All families with students within the Level 1 proficiency status, will be a offered to be a part of this program. These educational videos will be specially chosen by the English Learner Teacher and will be sent home regularly to watch. The videos are to help the child develop early reading skills.\r\n\r\nParents that do not have access to a dvd player will have the opportunity to check out a dvd player to use for the year. The plan is to use these videos and educational dvd's for the years to come for other EL students.\r\nnannan

==========

The 51 fifth grade students that will cycle through my classroom this year all love learning, at least most of the time. At our school, 97.3% of the students receive free or reduced price lunch. Of the 560 students, 97.3% are minority students. \r\nThe school has a vibrant community that loves to get together and celebrate. Around Halloween there is a whole school parade to show off the beautiful costumes that students wear. On Cinco de Mayo we put on a big festival with crafts made by the students, dances, and games. At the end of the year the school hosts a carnival to celebrate the hard work put in during the school year, with a dunk tank being the most popular activity.My students will use these five brightly colored Hokki stools in place of regular, stationary, 4-legged chairs. As I will only have a total of ten in the classroom and not enough for each student to have an individual one, they will be used in a variety of ways. During independent reading time they will be used as special chairs students will each use on occasion. I will utilize them in place of chairs at my small group tables during math and reading times. The rest of the day they will be used by the students who need the highest amount of movement in their life in order to stay focused on school.\r\n\r\nWhenever asked what at the classroom is missing, my students always say more Hokki Stools. They can't get their fill of the 5 stools we already have. When the students are sitting in group with me on the Hokki Stools, they are always moving, but at the same time doing their work. Anytime the students get to pick where they can sit, the Hokki Stools are the first to be taken. There are always students who head over to the kidney table to get one of the stools who are disappointed as there are not enough of them. \r\n\r\nWe ask a lot of students to sit for 7 hours a day. The Hokki stools will be a compromise that allow my students to do desk work and move at the same time. These stools will help students to meet their 60 minutes a day of movement by allowing them to activate their core muscles for balance while they sit. For many of my students, these chairs will take away the barrier that exists in schools for a child who can't sit still.nannan

==========

How do you remember your days of school? Was it in a sterile environment with plain walls, rows of desks, and a teacher in front of the room? A typical day in our room is nothing like that. I work hard to create a warm inviting theme

d room for my students look forward to coming to each day.\r\n\r\nMy class is made up of 28 wonderfully unique boys and girls of mixed races in Arkansas.\r\nThey attend a Title I school, which means there is a high enough percentage of free and reduced-price lunch to qualify. Our school is an \"open classroom\" concept, which is very unique as there are no walls separating the classrooms. These 9 and 10 year-old students are very eager learners; they are like sponges, absorbing all the information and experiences and keep on wanting more.With these resources such as the comfy red throw pillows and the whimsical nautical hanging decor and the blue fish nets, I will be able to help create the mood in our classroom setting to be one of a themed nautical environment. Creating a classroom environment is very important in the success in each and every child's education. The nautical photo props will be used with each child as they step foot into our classroom for the first time on Meet the Teacher evening. I'll take pictures of each child with them, have them developed, and then hung in our classroom ready for their first day of 4th grade. This kind gesture will set the tone before even the first day of school! The nautical thank you cards will be used throughout the year by the students as they create thank you cards to their team groups.\r\n\r\nYour generous donations will help me to help make our classroom a fun, inviting, learning environment from day one.\r\n\r\nIt costs lost of money out of my own pocket on resources to get our classroom ready. Please consider helping with this project to make our new school year a very successful one. Thank you!nannan

==========

My kindergarten students have varied disabilities ranging from speech and language delays, cognitive delays, gross/fine motor delays, to autism. They are eager beavers and always strive to work their hardest working past their limitations. \r\n\r\nThe materials we have are the ones I seek out for my students. I teach in a Title I school where most of the students receive free or reduced price lunch. Despite their disabilities and limitations, my students love coming to school and come eager to learn and explore.Have you ever felt like you had ants in your pants and you needed to groove and move as you were in a meeting? This is how my kids feel all the time. The want to be able to move as they learn or so they say.Wobble chairs are the answer and I love then because they develop their core, which enhances gross motor and in Turn fine motor skills. \r\nThey also want to learn through games, my kids don't want to sit and do worksheets. They want to learn to count by jumping and playing. Physical engagement is the key to our success. The number toss and color and shape mats can make that happen. My students will forget they are doing work and just have the fun a 6 year old deserves.nannan

==========

The mediocre teacher tells. The good teacher explains. The superior teacher demonstrates. The great teacher inspires. -William A. Ward\r\n\r\nMy school has 803 students which is makeup is 97.6% African-American, making up the largest segment of the student body. A typical school in Dallas is made up of 23.2% African-American students. Most of the students are on free or reduced lunch. We aren't receiving doctors, lawyers, or engineers children from rich backgrounds or neighborhoods. As an educator I am inspiring minds of young children and we focus not only on academics but one smart, effective, efficient, and disciplined students with good character.In our classroom we can utilize the Bluetooth for swift transitions during class. I use a speaker which doesn't amplify the sound enough to receive the message. Due to the volume of my speaker my students can't hear videos or books clearly and it isn't making the lessons as meaningful. But with the bluetooth speaker my students will be able to hear and I can stop, pause and replay it at any time.\r\nThe cart will allow me to have more room for storage of things that are needed for the day and has an extra part to it I can use. The table top chart has all of the letter, words and pictures for students to learn about different letters and it

```
is more accessible.nannan
==================================================
```

In [17]:
```
# https://stackoverflow.com/a/47091490/4084039
import re

def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)

    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    return phrase
```

In [18]:
```
sent = decontracted(project_data['essay'].values[20000])
print(sent)
print("="*50)
```

```
My kindergarten students have varied disabilities ranging from speech and lan
guage delays, cognitive delays, gross/fine motor delays, to autism. They are
eager beavers and always strive to work their hardest working past their limi
tations. \r\n\r\nThe materials we have are the ones I seek out for my student
s. I teach in a Title I school where most of the students receive free or red
uced price lunch.  Despite their disabilities and limitations, my students lo
ve coming to school and come eager to learn and explore.Have you ever felt li
ke you had ants in your pants and you needed to groove and move as you were i
n a meeting? This is how my kids feel all the time. The want to be able to mo
ve as they learn or so they say.Wobble chairs are the answer and I love then
because they develop their core, which enhances gross motor and in Turn fine
motor skills. \r\nThey also want to learn through games, my kids do not want
to sit and do worksheets. They want to learn to count by jumping and playing.
Physical engagement is the key to our success. The number toss and color and
shape mats can make that happen. My students will forget they are doing work
and just have the fun a 6 year old deserves.nannan
==================================================
```

In [19]:
```
# \r \n \t remove from string python: http://texthandler.com/info/remove-line-breaks-python/
sent = sent.replace('\\r', ' ')
sent = sent.replace('\\"', ' ')
sent = sent.replace('\\n', ' ')
print(sent)
```

```
My kindergarten students have varied disabilities ranging from speech and lan
guage delays, cognitive delays, gross/fine motor delays, to autism. They are
eager beavers and always strive to work their hardest working past their limi
tations.     The materials we have are the ones I seek out for my students. I
teach in a Title I school where most of the students receive free or reduced
price lunch.  Despite their disabilities and limitations, my students love co
ming to school and come eager to learn and explore.Have you ever felt like yo
u had ants in your pants and you needed to groove and move as you were in a m
eeting? This is how my kids feel all the time. The want to be able to move as
they learn or so they say.Wobble chairs are the answer and I love then becaus
e they develop their core, which enhances gross motor and in Turn fine motor
skills.   They also want to learn through games, my kids do not want to sit a
nd do worksheets. They want to learn to count by jumping and playing. Physica
l engagement is the key to our success. The number toss and color and shape m
ats can make that happen. My students will forget they are doing work and jus
t have the fun a 6 year old deserves.nannan
```

In [20]:
```
#remove spacial character: https://stackoverflow.com/a/5843547/4084039
sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
print(sent)
```

```
My kindergarten students have varied disabilities ranging from speech and lan
guage delays cognitive delays gross fine motor delays to autism They are eage
r beavers and always strive to work their hardest working past their limitati
ons The materials we have are the ones I seek out for my students I teach in
a Title I school where most of the students receive free or reduced price lun
ch Despite their disabilities and limitations my students love coming to scho
ol and come eager to learn and explore Have you ever felt like you had ants i
n your pants and you needed to groove and move as you were in a meeting This
is how my kids feel all the time The want to be able to move as they learn or
so they say Wobble chairs are the answer and I love then because they develop
their core which enhances gross motor and in Turn fine motor skills They also
want to learn through games my kids do not want to sit and do worksheets They
want to learn to count by jumping and playing Physical engagement is the key
to our success The number toss and color and shape mats can make that happen
My students will forget they are doing work and just have the fun a 6 year ol
d deserves nannan
```

In [21]:
```
# https://gist.github.com/sebleier/554280
# we are removing the words from the stop words list: 'no', 'nor', 'not'
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you'
, "you're", "you've",\
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he'
, 'him', 'his', 'himself', \
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'it
self', 'they', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 't
hat', "that'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have',
'has', 'had', 'having', 'do', 'does', \
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'becau
se', 'as', 'until', 'while', 'of', \
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into',
'through', 'during', 'before', 'after',\
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on',
'off', 'over', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'a
ll', 'any', 'both', 'each', 'few', 'more',\
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'tha
n', 'too', 'very', \
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "shoul
d've", 'now', 'd', 'll', 'm', 'o', 're', \
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn',
"didn't", 'doesn', "doesn't", 'hadn',\
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'm
a', 'mightn', "mightn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shoul
dn't", 'wasn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't"]
```

In [22]:
```
# Combining all the above stundents
from tqdm import tqdm
preprocessed_essays = []
# tqdm is for printing the status bar
for sentance in tqdm(project_data['essay'].values):
    sent = decontracted(sentance)
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
    # https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split() if e not in stopwords)
    preprocessed_essays.append(sent.lower().strip())
```

```
100%|██████████████████████████████████████████████████████████████████████████
| 109248/109248 [01:00<00:00, 1802.70it/s]
```

In [23]:
```
# after preprocesing
preprocessed_essays[20000]
```

Out[23]:
```
'my kindergarten students varied disabilities ranging speech language delays
cognitive delays gross fine motor delays autism they eager beavers always str
ive work hardest working past limitations the materials ones i seek students
i teach title i school students receive free reduced price lunch despite disa
bilities limitations students love coming school come eager learn explore hav
e ever felt like ants pants needed groove move meeting this kids feel time th
e want able move learn say wobble chairs answer i love develop core enhances
gross motor turn fine motor skills they also want learn games kids not want s
it worksheets they want learn count jumping playing physical engagement key s
uccess the number toss color shape mats make happen my students forget work f
un 6 year old deserves nannan'
```

1.7 Preprocessing of `project_title`

In [24]:
```
# similarly you can preprocess the titles also
project_data.head(2)
```

Out[24]:

	Unnamed: 0	id	teacher_id	school_state	project_submi
0	160221	p253737	c90749f5d961ff158d4b4d1e7dc665fc	IN	2016-12-05 13:
1	140945	p258326	897464ce9ddc600bced1151f324dd63a	FL	2016-10-25 09:

In [25]:
```
# printing some random essays.
print(project_data['project_title'].values[0])
print("="*50)
print(project_data['project_title'].values[150])
print("="*50)
print(project_data['project_title'].values[1000])
print("="*50)
print(project_data['project_title'].values[20000])
print("="*50)
print(project_data['project_title'].values[99999])
print("="*50)
```

```
Educational Support for English Learners at Home
==================================================
More Movement with Hokki Stools
==================================================
Sailing Into a Super 4th Grade Year
==================================================
We Need To Move It While We Input It!
==================================================
Inspiring Minds by Enhancing the Educational Experience
==================================================
```

In [26]:
```
sent_title = decontracted(project_data['project_title'].values[20000])
print(sent_title)
print("="*50)
```

```
We Need To Move It While We Input It!
==================================================
```

In [27]:
```
# \r \n \t remove from string python: http://texthandler.com/info/remove-line-breaks-python/
sent_title = sent_title.replace('\\r', ' ')
sent_title = sent_title.replace('\\"', ' ')
sent_title = sent_title.replace('\\n', ' ')
print(sent_title)
```

```
We Need To Move It While We Input It!
```

In [28]:
```
#remove spacial character: https://stackoverflow.com/a/5843547/4084039
sent_title = re.sub('[^A-Za-z0-9]+', ' ', sent_title)
print(sent_title)
```

```
We Need To Move It While We Input It
```

In [29]:
```
# Combining all the above statemennts
from tqdm import tqdm
preprocessed_title = []
# tqdm is for printing the status bar
for sentance in tqdm(project_data['project_title'].values):
    sent_title = decontracted(sentance)
    sent_title = sent_title.replace('\\r', ' ')
    sent_title = sent_title.replace('\\"', ' ')
    sent_title = sent_title.replace('\\n', ' ')
    sent_title = re.sub('[^A-Za-z0-9]+', ' ', sent_title)
    # https://gist.github.com/sebleier/554280
    sent_title = ' '.join(e for e in sent_title.split() if e not in stopwords)
    preprocessed_title.append(sent_title.lower().strip())
```

```
100%|██████████████████████████████████████████████████████████████████████████████| 
109248/109248 [00:02<00:00, 39963.57it/s]
```

In [30]:
```
# after preprocesing
preprocessed_title[10]
```

Out[30]: 'reading changes lives'

In [31]:
```
# Combining all the above statemennts
from tqdm import tqdm
preprocessed_prj_sum = []
# tqdm is for printing the status bar
for sentance in tqdm(project_data['project_resource_summary'].values):
    sent_title = decontracted(sentance)
    sent_title = sent_title.replace('\\r', ' ')
    sent_title = sent_title.replace('\\"', ' ')
    sent_title = sent_title.replace('\\n', ' ')
    sent_title = re.sub('[^A-Za-z0-9]+', ' ', sent_title)
    # https://gist.github.com/sebleier/554280
    sent_title = ' '.join(e for e in sent_title.split() if e not in stopwords)
    preprocessed_prj_sum.append(sent_title.lower().strip())
```

```
100%|██████████████████████████████████████████████████████████████████████████████| 
109248/109248 [00:06<00:00, 16680.32it/s]
```

1.9 Preparing data for models

In [38]:
```
project_data.columns
```

Out[38]:
```
Index(['Unnamed: 0', 'id', 'teacher_id', 'school_state',
       'project_submitted_datetime', 'project_title', 'project_essay_1',
       'project_essay_2', 'project_essay_3', 'project_essay_4',
       'project_resource_summary',
       'teacher_number_of_previously_posted_projects', 'project_is_approved',
       'clean_categories', 'clean_subcategories', 'clean_grade',
       'clean_tea_pfx', 'essay'],
      dtype='object')
```

we are going to consider

- school_state : categorical data
- clean_categories : categorical data
- clean_subcategories : categorical data
- project_grade_category : categorical data
- teacher_prefix : categorical data

- project_title : text data
- text : text data
- project_resource_summary: text data (optinal)

- quantity : numerical (optinal)
- teacher_number_of_previously_posted_projects : numerical
- price : numerical

Using Pretrained Models: Avg W2V

In [39]:
```
'''
# Reading glove vectors in python: https://stackoverflow.com/a/38230349/408403
9
def loadGloveModel(gloveFile):
    print ("Loading Glove Model")
    f = open(gloveFile,'r', encoding="utf8")
    model = {}
    for line in tqdm(f):
        splitLine = line.split()
        word = splitLine[0]
        embedding = np.array([float(val) for val in splitLine[1:]])
        model[word] = embedding
    print ("Done.",len(model)," words loaded!")
    return model
model = loadGloveModel('glove.42B.300d.txt')

# ============================
Output:

Loading Glove Model
1917495it [06:32, 4879.69it/s]
Done. 1917495  words loaded!

# ============================

words = []
for i in preproced_texts:
    words.extend(i.split(' '))

for i in preproced_titles:
    words.extend(i.split(' '))
print("all the words in the coupus", len(words))
words = set(words)
print("the unique words in the coupus", len(words))

inter_words = set(model.keys()).intersection(words)
print("The number of words that are present in both glove vectors and our coup
us", \
      len(inter_words),"(",np.round(len(inter_words)/len(words)*100,3),"%)")

words_courpus = {}
words_glove = set(model.keys())
for i in words:
    if i in words_glove:
        words_courpus[i] = model[i]
print("word 2 vec length", len(words_courpus))


# stronging variables into pickle files python: http://www.jessicayung.com/how
-to-use-pickle-to-save-and-load-variables-in-python/

import pickle
with open('glove_vectors', 'wb') as f:
    pickle.dump(words_courpus, f)
```

```
'''
```

Out[39]:
```
'\n# Reading glove vectors in python: https://stackoverflow.com/a/38230349/40
84039\ndef loadGloveModel(gloveFile):\n    print ("Loading Glove Model")\n
f = open(gloveFile,\'r\', encoding="utf8")\n    model = {}\n    for line in t
qdm(f):\n        splitLine = line.split()\n        word = splitLine[0]\n
embedding = np.array([float(val) for val in splitLine[1:]])\n        model[wo
rd] = embedding\n    print ("Done.",len(model)," words loaded!")\n    return
model\nmodel = loadGloveModel(\'glove.42B.300d.txt\')\n\n# ==================
==========\nOutput:\n    \nLoading Glove Model\n1917495it [06:32, 4879.69it/
s]\nDone. 1917495  words loaded!\n\n# ============================\n\nwords =
[]\nfor i in preproced_texts:\n    words.extend(i.split(\' \'))\n\nfor i in p
reproced_titles:\n    words.extend(i.split(\' \'))\nprint("all the words in t
he coupus", len(words))\nwords = set(words)\nprint("the unique words in the c
oupus", len(words))\n\ninter_words = set(model.keys()).intersection(words)\np
rint("The number of words that are present in both glove vectors and our coup
us",       len(inter_words),"(",np.round(len(inter_words)/len(words)*100,
3),"%)")\n\nwords_courpus = {}\nwords_glove = set(model.keys())\nfor i in wor
ds:\n    if i in words_glove:\n        words_courpus[i] = model[i]\nprint("wo
rd 2 vec length", len(words_courpus))\n\n\n# stronging variables into pickle
files python: http://www.jessicayung.com/how-to-use-pickle-to-save-and-load-v
ariables-in-python/\n\nimport pickle\nwith open(\'glove_vectors\', \'wb\') as
f:\n    pickle.dump(words_courpus, f)\n\n\n'
```

In [40]:
```
# stronging variables into pickle files python: http://www.jessicayung.com/how
-to-use-pickle-to-save-and-load-variables-in-python/
# make sure you have the glove_vectors file
with open('glove_vectors', 'rb') as f:
    model = pickle.load(f)
    glove_words =  set(model.keys())
```

Vectorizing Numerical features

In [41]:
```
#print(type(project_data))
#print(type(price_data))
```

In [42]:
```
price_data = resource_data.groupby('id').agg({'price':'sum', 'quantity':'sum'
}).reset_index()
project_data = pd.merge(project_data, price_data, on='id', how='left')

print("Number of data points in train data", project_data.shape)
print('-'*50)
print("The attributes of data :", project_data.columns.values)
```

```
Number of data points in train data (109248, 20)
--------------------------------------------------
The attributes of data : ['Unnamed: 0' 'id' 'teacher_id' 'school_state'
 'project_submitted_datetime' 'project_title' 'project_essay_1'
 'project_essay_2' 'project_essay_3' 'project_essay_4'
 'project_resource_summary' 'teacher_number_of_previously_posted_projects'
 'project_is_approved' 'clean_categories' 'clean_subcategories'
 'clean_grade' 'clean_tea_pfx' 'essay' 'price' 'quantity']
```

Adding word count and length column

In [43]:
```
#project_data['essay_wc'] = preprocessed_essays_wc
#project_data['essay_len'] = preprocessed_essays_len
#
#project_data['title_wc'] = preprocessed_title_wc
#project_data['title_len'] = preprocessed_title_len
#
#project_data['prj_res_sum_wc'] = preprocessed_prj_sum_wc
#project_data['prj_res_sum_len'] = preprocessed_prj_sum_len
#
#print("Number of data points in train data", project_data.shape)
#print('-'*50)
#print("The attributes of data :", project_data.columns.values)
```

Assignment 9: RF and GBDT

Response Coding: Example

The response tabel is built only on train dataset. For a category which is not there in train data and present in test data, we will encode them with default values Ex: in our test data if have State: D then we encode it as [0.5, 0.05]

1. **Apply both Random Forrest and GBDT on these feature sets**

    - Set 1: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(BOW) + preprocessed_eassay (BOW)
    - Set 2: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(TFIDF)+ preprocessed_eassay (TFIDF)
    - Set 3: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(AVG W2V)+ preprocessed_eassay (AVG W2V)
    - Set 4: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(TFIDF W2V)+ preprocessed_eassay (TFIDF W2V)

2. **The hyper paramter tuning (Consider any two hyper parameters preferably n_estimators, max_depth)**

    - Find the best hyper parameter which will give the maximum AUC (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/receiver-operating-characteristic-curve-roc-curve-and-auc-1/) value
    - find the best hyper paramter using k-fold cross validation/simple cross validation data
    - use gridsearch cv or randomsearch cv or you can write your own for loops to do this task

3. **Representation of results**

    - You need to plot the performance of model both on train data and cross validation data for each hyper parameter, like shown in the figure
    with X-axis as **n_estimators**, Y-axis as **max_depth**, and Z-axis as **AUC Score** , we have given the notebook which explains how to plot this 3d plot, you can find it in the same drive *3d_scatter_plot.ipynb*

or

    - You need to plot the performance of model both on train data and cross validation data for each hyper parameter, like shown in the figure
    seaborn heat maps (https://seaborn.pydata.org/generated/seaborn.heatmap.html) with rows as **n_estimators**, columns as **max_depth**, and values inside the cell representing **AUC Score**
    - You can choose either of the plotting techniques: 3d plot or heat map
    - Once after you found the best hyper parameter, you need to train your model with it, and find the AUC on test data and plot the ROC curve on both train and test.
    Along with plotting ROC curve, you need to print the confusion matrix (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/confusion-matrix-tpr-fpr-fnr-tnr-1/) with predicted and original labels of test data points

4. **Conclusion**

    - You need to summarize the results at the end of the notebook, summarize it in the table format. To print out a table please refer to this prettytable library link (http://zetcode.com/python/prettytable/)

Note: Data Leakage

1. There will be an issue of data-leakage if you vectorize the entire data and then split it into train/cv/test.
2. To avoid the issue of data-leakag, make sure to split your data first and then vectorize it.
3. While vectorizing your data, apply the method fit_transform() on you train data, and apply the method transform() on cv/test data.
4. For more details please go through this link. (https://soundcloud.com/applied-ai-course/leakage-bow-and-tfidf)

2. Random Forest and GBDT

In [44]:
```
##taking 50K datapoint
project_data50K=project_data[:50000]
#project_data100K=project_data[:100000]
X=project_data50K
#X=project_data100K
print(project_data50K.shape)
#print(project_data100K.shape)
print(X.shape)
```

```
(50000, 20)
(50000, 20)
```

In [45]:
```
# makins Xi as 19 column matrix, where we create the modle and Yi as single co
lum matrix as a class label.
y = project_data50K['project_is_approved'].values
#project_data50K.drop(['project_is_approved'], axis=1, inplace=True)
#rint(y.shape)
#(project_data50K.head(1)
print(project_data50K.columns)
y50K=y[:50000]
y=y50K

#y = project_data['project_is_approved'].values
#project_data.drop(['project_is_approved'], axis=1, inplace=True)
##print(y.shape)
#project_data.head(1)
#
#y100K=y[:100000]
#y=y100K

#y = project_data['project_is_approved'].values
#project_data.drop(['project_is_approved'], axis=1, inplace=True)
#print(y.shape)
#project_data.head(1)
```

```
Index(['Unnamed: 0', 'id', 'teacher_id', 'school_state',
       'project_submitted_datetime', 'project_title', 'project_essay_1',
       'project_essay_2', 'project_essay_3', 'project_essay_4',
       'project_resource_summary',
       'teacher_number_of_previously_posted_projects', 'project_is_approved',
       'clean_categories', 'clean_subcategories', 'clean_grade',
       'clean_tea_pfx', 'essay', 'price', 'quantity'],
      dtype='object')
```

In [46]:
```
print(X.shape)
print(y.shape)
```

```
(50000, 20)
(50000,)
```

2.1 Splitting data into Train and cross validation(or test): Stratified Sampling

In [47]:
```
# please write all the code with proper documentation, and proper titles for e
ach subsection
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debug
ging your code
# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the rea
der
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

In [48]:
```
# train test split | https://scikit-learn.org/stable/modules/generated/sklear
n.model_selection.train_test_split.html
# spliting Xq and Yq in Train(further into Train and CV) and Test matrix
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, stra
tify=y)
#X_train, X_cv, y_train, y_cv = train_test_split(X_train, y_train, test_size=
0.33, stratify=y_train)

print(X_train.shape, y_train.shape)
#print(X_cv.shape, y_cv.shape)
print(X_test.shape, y_test.shape)

print("="*100)
```

```
(33500, 20) (33500,)
(16500, 20) (16500,)
====================================================================================
======================
```

In [49]:
```
X_train.columns
```

Out[49]:
```
Index(['Unnamed: 0', 'id', 'teacher_id', 'school_state',
       'project_submitted_datetime', 'project_title', 'project_essay_1',
       'project_essay_2', 'project_essay_3', 'project_essay_4',
       'project_resource_summary',
       'teacher_number_of_previously_posted_projects', 'project_is_approved',
       'clean_categories', 'clean_subcategories', 'clean_grade',
       'clean_tea_pfx', 'essay', 'price', 'quantity'],
      dtype='object')
```

2.1.1 Make Data Model Ready: Response encoding school_state categorical data

In [295]:
```
# using value_count() to get each category's count, when project is Approved
X_train_school_st_One = X_train["school_state"][X_train['project_is_approved']
==1]
X_train_school_st_One = X_train_school_st_One.value_counts()
X_test_school_st_One = X_test["school_state"][X_test['project_is_approved']==1
]
X_test_school_st_One = X_test_school_st_One.value_counts()

# using value_count() to get each category's count, when project is Not Approv
ed
X_train_school_st_Zero = X_train["school_state"][X_train['project_is_approved'
]==0]
X_train_school_st_Zero = X_train_school_st_Zero.value_counts()
X_test_school_st_Zero = X_test["school_state"][X_test['project_is_approved']==
0]
X_test_school_st_Zero = X_test_school_st_Zero.value_counts()

# using value_count() to get each category's count
X_train_school_st_All = X_train["school_state"]
X_train_school_st_All = X_train_school_st_All.value_counts()
X_test_school_st_All = X_test["school_state"]
X_test_school_st_All = X_test_school_st_All.value_counts()

# Creating a Dataframe, having response Table for Category, coulumn are derive
d above
X_train_response_tab_School= pd.DataFrame(data=dict(X_train_school_st_One=X_tr
ain_school_st_One,X_train_school_st_Zero=X_train_school_st_Zero,X_train_school
_st_All=X_train_school_st_All))
#X_test_response_tab_School= pd.DataFrame(data=dict(X_test_school_st_One=X_tes
t_school_st_One,X_test_school_st_Zero=X_test_school_st_Zero,X_test_school_st_A
ll=X_test_school_st_All))

# Adding two more coulmn, which are calculated as per the probablity i.e. Tota
l given category count per state/Total category count
X_train_response_tab_School["State_0"]=X_train_response_tab_School["X_train_sc
hool_st_Zero"]/X_train_response_tab_School["X_train_school_st_All"]
X_train_response_tab_School["State_1"]=X_train_response_tab_School["X_train_sc
hool_st_One"]/X_train_response_tab_School["X_train_school_st_All"]
#X_test_response_tab_School["State_0"]=X_test_response_tab_School["X_test_scho
ol_st_Zero"]/X_test_response_tab_School["X_test_school_st_All"]
#X_test_response_tab_School["State_1"]=X_test_response_tab_School["X_test_scho
ol_st_One"]/X_test_response_tab_School["X_test_school_st_All"]

# resetting an index,
X_train_response_tab_School=X_train_response_tab_School.reset_index()
#X_test_response_tab_School=X_test_response_tab_School.reset_index()

# renaming an column header
X_train_response_tab_School.columns=['school_state','X_train_school_st_One','X
_train_school_st_Zero','X_train_school_st_All','State_0','State_1']
print(X_train_response_tab_School.shape)
X_train_response_tab_School.head(5)

#X_test_response_tab_School.columns=['school_state','X_test_school_st_One','X_
test_school_st_Zero','X_test_school_st_All','State_0','State_1']
```

```
#print(X_test_response_tab_School.shape)
#X_test_response_tab_School
```

```
(51, 6)
```

Out[295]:

	school_state	X_train_school_st_One	X_train_school_st_Zero	X_train_school_st_All
0	AK	85	18	103
1	AL	449	79	528
2	AR	251	49	300
3	AZ	559	100	659
4	CA	4024	674	4698

In [298]:
```
#(X_train_response_tab_School.isna())
#X_train_response_tab_School[:].isnull()
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Since thjis is the Response table, we find from Train data, and we need to u
se the same in Test data,
# we deleting rows whcih are nan
delete_index=[]
for i in range(X_train_response_tab_School.shape[0]):
    if X_train_response_tab_School['X_train_school_st_One'].isnull().iloc[i]:
        delete_index.append(i)
for i in range(X_train_response_tab_School.shape[0]):
    if X_train_response_tab_School['X_train_school_st_Zero'].isnull().iloc[i]:
        delete_index.append(i)
#print(delete_index)

X_train_response_tab_School.dropna
print(X_train_response_tab_School.shape)
```

```
(51, 6)
```

In [299]:
```
X_train_response_tab_School=X_train_response_tab_School.dropna()
X_train_response_tab_School.head(5)
```

Out[299]:

	school_state	X_train_school_st_One	X_train_school_st_Zero	X_train_school_st_All
0	AK	85	18	103
1	AL	449	79	528
2	AR	251	49	300
3	AZ	559	100	659
4	CA	4024	674	4698

In [300]:
```
# create a new Dataframe for X_train_school
X_train_school=pd.DataFrame(data=dict(school_state=X_train["school_state"],pro
ject_is_approved=X_train["project_is_approved"]))
X_train_school

# create a new Dataframe for X_test_school
X_test_school=pd.DataFrame(data=dict(school_state=X_test["school_state"],proje
ct_is_approved=X_test["project_is_approved"]))
X_test_school.head(5)
```

Out[300]:

	school_state	project_is_approved
31254	GA	1
35307	OK	1
23220	IL	1
34585	CA	1
46330	PA	1

In [301]:
```
# join X_train_school with response table we created earlier
X_train_school=pd.merge(X_train_school,X_train_response_tab_School, on='school
_state',how='left')
print(X_train_school.shape)
X_train_school

# join X_test_school with response table we created earlier
# NOTE: we are tallying x_test with response table whihc we get from x_train
X_test_school=pd.merge(X_test_school,X_train_response_tab_School, on='school_s
tate',how='left')
print(X_test_school.shape)
X_test_school.head(5)
```

```
(33500, 7)
(16500, 7)
```

Out[301]:

	school_state	project_is_approved	X_train_school_st_One	X_train_school_st_Zero	X
0	GA	1	995	195	1
1	OK	1	612	114	7
2	IL	1	1143	199	1
3	CA	1	4024	674	4
4	PA	1	812	138	9

In [302]:
```
# dropping unwanted column names
X_train_school.drop(['school_state','project_is_approved','X_train_school_st_One','X_train_school_st_Zero','X_train_school_st_All'], axis=1, inplace=True)
print(X_train_school.shape)
X_train_school

# dropping unwanted column names
X_test_school.drop(['school_state','project_is_approved','X_train_school_st_One','X_train_school_st_Zero','X_train_school_st_All'], axis=1, inplace=True)
print(X_test_school.shape)
X_test_school.head(5)
```

```
(33500, 2)
(16500, 2)
```

Out[302]:

	State_0	State_1
0	0.163866	0.836134
1	0.157025	0.842975
2	0.148286	0.851714
3	0.143465	0.856535
4	0.145263	0.854737

In [56]:
```
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with-nan-in-pandas-dataframe
# Checking if there is nay nan in Test dataframe
nan_index_tr=[]
for i in range(X_train_school.shape[0]):
    if X_train_school['State_0'].isnull().iloc[i]:
        nan_index_tr.append(i)
for i in range(X_train_school.shape[0]):
    if X_train_school['State_1'].isnull().iloc[i]:
        nan_index_tr.append(i)
print(nan_index_tr)

nan_index_te=[]
for i in range(X_test_school.shape[0]):
    if X_test_school['State_0'].isnull().iloc[i]:
        nan_index_te.append(i)
for i in range(X_test_school.shape[0]):
    if X_test_school['State_1'].isnull().iloc[i]:
        nan_index_te.append(i)
print(nan_index_te)
```

```
[]
[]
```

In [303]:
```
# roundoff all element in Dataframe | https://www.geeksforgeeks.org/python-pandas-dataframe-round/
X_train_school.round(2)
X_test_school.round(2).head(5)
```

Out[303]:

	State_0	State_1
0	0.16	0.84
1	0.16	0.84
2	0.15	0.85
3	0.14	0.86
4	0.15	0.85

In [58]:
```
print(type(X_train_school))
print(type(X_test_school))
print(X_train_school.shape)
print(X_test_school.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
(33500, 2)
(16500, 2)
```

2.1.2 Make Data Model Ready: Response encoding clean_categories

In [304]:
```
# using value_count() to get each category's count, whene project is Approved
X_train_Clean_category_One = X_train["clean_categories"][X_train['project_is_a
pproved']==1]
X_train_Clean_category_One = X_train_Clean_category_One.value_counts()
X_test_Clean_category_One = X_test["clean_categories"][X_test['project_is_appr
oved']==1]
X_test_Clean_category_One = X_test_Clean_category_One.value_counts()

# using value_count() to get each category's count, whene project is Not Appro
ved
X_train_Clean_category_Zero = X_train["clean_categories"][X_train['project_is_
approved']==0]
X_train_Clean_category_Zero = X_train_Clean_category_Zero.value_counts()
X_test_Clean_category_Zero = X_test["clean_categories"][X_test['project_is_app
roved']==0]
X_test_Clean_category_Zero = X_test_Clean_category_Zero.value_counts()

# using value_count() to get each category's count
X_train_Clean_category_All = X_train["clean_categories"]
X_train_Clean_category_All = X_train_Clean_category_All.value_counts()
X_test_Clean_category_All = X_test["clean_categories"]
X_test_Clean_category_All = X_test_Clean_category_All.value_counts()

# Creating a Dataframe, having response Table for Category, coulumn are derive
d above
X_train_response_tab_category= pd.DataFrame(data=dict(X_train_Clean_category_O
ne=X_train_Clean_category_One,X_train_Clean_category_Zero=X_train_Clean_catego
ry_Zero,X_train_Clean_category_All=X_train_Clean_category_All))
#X_test_response_tab_category= pd.DataFrame(data=dict(X_test_Clean_category_On
e=X_test_Clean_category_One,X_test_Clean_category_Zero=X_test_Clean_category_Z
ero,X_test_Clean_category_All=X_test_Clean_category_All))

# Adding two more coulmn, which are calculated as per the probablity i.e. Tota
l given category count per state/Total category count
X_train_response_tab_category["State_0"]=X_train_response_tab_category["X_trai
n_Clean_category_Zero"]/X_train_response_tab_category["X_train_Clean_category_
All"]
X_train_response_tab_category["State_1"]=X_train_response_tab_category["X_trai
n_Clean_category_One"]/X_train_response_tab_category["X_train_Clean_category_A
ll"]
#X_test_response_tab_category["State_0"]=X_test_response_tab_category["X_test_
Clean_category_Zero"]/X_test_response_tab_category["X_test_Clean_category_Al
l"]
#X_test_response_tab_category["State_1"]=X_test_response_tab_category["X_test_
Clean_category_One"]/X_test_response_tab_category["X_test_Clean_category_All"]

# resetting an index,
X_train_response_tab_category=X_train_response_tab_category.reset_index()
#X_test_response_tab_category=X_test_response_tab_category.reset_index()

# renaming an column header
X_train_response_tab_category.columns=['clean_categories','X_train_Clean_categ
ory_One','X_train_Clean_category_Zero','X_train_Clean_category_All','State_0',
'State_1']
print(X_train_response_tab_category.shape)
X_train_response_tab_category.head(5)
```

```
#X_test_response_tab_category.columns=['clean_categories','X_test_Clean_catego
ry_One','X_test_Clean_category_Zero','X_test_Clean_category_All','State_0','St
ate_1']
#print(X_test_response_tab_category.shape)
#X_test_response_tab_category
```

```
(50, 6)
```

Out[304]:

	clean_categories	X_train_Clean_category_One	X_train_Clean_category_Zero	X_trai
0	AppliedLearning	919.0	219.0	1138
1	AppliedLearning Health_Sports	154.0	26.0	180
2	AppliedLearning History_Civics	45.0	7.0	52
3	AppliedLearning Literacy_Language	561.0	102.0	663
4	AppliedLearning Math_Science	260.0	71.0	331

In [306]:
```
#(X_train_response_tab_category.isna())
#X_train_response_tab_category[:].isnull()
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Since thjis is the Response table, we find from Train data, and we need to u
se the same in Test data,
# we deleting rows whcih are nan
delete_index=[]
for i in range(X_train_response_tab_category.shape[0]):
    if X_train_response_tab_category['X_train_Clean_category_One'].isnull().il
oc[i]:
        delete_index.append(i)
for i in range(X_train_response_tab_category.shape[0]):
    if X_train_response_tab_category['X_train_Clean_category_Zero'].isnull().i
loc[i]:
        delete_index.append(i)
print(delete_index)

X_train_response_tab_category.dropna
X_train_response_tab_category.head(5)
```

```
[44, 7, 15, 18, 30]
```

Out[306]:

	clean_categories	X_train_Clean_category_One	X_train_Clean_category_Zero	X_trai
0	AppliedLearning	919.0	219.0	1138
1	AppliedLearning Health_Sports	154.0	26.0	180
2	AppliedLearning History_Civics	45.0	7.0	52
3	AppliedLearning Literacy_Language	561.0	102.0	663
4	AppliedLearning Math_Science	260.0	71.0	331

In [307]:
```
X_train_response_tab_category=X_train_response_tab_category.dropna()
X_train_response_tab_category.head(5)
```

Out[307]:

	clean_categories	X_train_Clean_category_One	X_train_Clean_category_Zero	X_trai
0	AppliedLearning	919.0	219.0	1138
1	AppliedLearning Health_Sports	154.0	26.0	180
2	AppliedLearning History_Civics	45.0	7.0	52
3	AppliedLearning Literacy_Language	561.0	102.0	663
4	AppliedLearning Math_Science	260.0	71.0	331

In [308]:
```
# create a new Dataframe for X_train_category
X_train_category=pd.DataFrame(data=dict(clean_categories=X_train["clean_categories"],project_is_approved=X_train["project_is_approved"]))
X_train_category

# create a new Dataframe for X_test_category
X_test_category=pd.DataFrame(data=dict(clean_categories=X_test["clean_categories"],project_is_approved=X_test["project_is_approved"]))
X_test_category.head(5)
```

Out[308]:

	clean_categories	project_is_approved
31254	Math_Science	1
35307	SpecialNeeds	1
23220	Health_Sports	1
34585	Math_Science AppliedLearning	1
46330	AppliedLearning Literacy_Language	1

In [309]:
```
# join X_train_category with response table we created earlier
X_train_category=pd.merge(X_train_category,X_train_response_tab_category, on=
'clean_categories',how='left')
print(X_train_category.shape)
X_train_category

# join X_test_category with response table we created earlier
# NOTE: we are tallying x_test with response table whihc we get from x_train
X_test_category=pd.merge(X_test_category,X_train_response_tab_category, on='cl
ean_categories',how='left')
print(X_test_category.shape)
X_test_category.head(5)
```

```
(33500, 7)
(16500, 7)
```

Out[309]:

	clean_categories	project_is_approved	X_train_Clean_category_One	X_train_Clean_
0	Math_Science	1	4259.0	948.0
1	SpecialNeeds	1	1034.0	255.0
2	Health_Sports	1	2658.0	480.0
3	Math_Science AppliedLearning	1	292.0	56.0
4	AppliedLearning Literacy_Language	1	561.0	102.0

In [310]:
```
# dropping unwanted column names
X_train_category.drop(['clean_categories','project_is_approved','X_train_Clean
_category_One','X_train_Clean_category_Zero','X_train_Clean_category_All'], ax
is=1, inplace=True)
print(X_train_category.shape)
X_train_category

# dropping unwanted column names
X_test_category.drop(['clean_categories','project_is_approved','X_train_Clean_
category_One','X_train_Clean_category_Zero','X_train_Clean_category_All'], axi
s=1, inplace=True)
print(X_test_category.shape)
X_test_category.head(5)
```

```
(33500, 2)
(16500, 2)
```

Out[310]:

	State_0	State_1
0	0.182063	0.817937
1	0.197828	0.802172
2	0.152964	0.847036
3	0.160920	0.839080
4	0.153846	0.846154

In [65]:
```
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Checking if there is nay nan in Test dataframe
nan_index_tr=[]
for i in range(X_train_category.shape[0]):
    if X_train_category['State_0'].isnull().iloc[i]:
        nan_index_tr.append(i)
for i in range(X_train_category.shape[0]):
    if X_train_category['State_1'].isnull().iloc[i]:
        nan_index_tr.append(i)
print(nan_index_tr)

nan_index_te=[]
for i in range(X_test_category.shape[0]):
    if X_test_category['State_0'].isnull().iloc[i]:
        nan_index_te.append(i)
for i in range(X_test_category.shape[0]):
    if X_test_category['State_1'].isnull().iloc[i]:
        nan_index_te.append(i)
print(nan_index_te)
```

```
[1555, 1961, 2028, 2631, 3804, 4137, 6119, 6440, 6664, 9178, 14395, 15532, 18
278, 20977, 25673, 26217, 26334, 29550, 32113, 33077, 1555, 1961, 2028, 2631,
3804, 4137, 6119, 6440, 6664, 9178, 14395, 15532, 18278, 20977, 25673, 26217,
26334, 29550, 32113, 33077]
[88, 3902, 4167, 4347, 6997, 8482, 10158, 12285, 13133, 15322, 15669, 16480,
88, 3902, 4167, 4347, 6997, 8482, 10158, 12285, 13133, 15322, 15669, 16480]
```

In [313]:
```
# For a category which is not there in train data and present in test data, we will
# encode them with default values Ex: in our test data if have State: D then we encode it as [0.5, 0.5]
X_train_category.iloc[nan_index_tr,[0,1]]=[0.5, 0.5]
X_train_category.iloc[nan_index_tr,[0,1]]

X_test_category.iloc[nan_index_te,[0,1]]=[0.5, 0.5]
X_test_category.iloc[nan_index_te,[0,1]]
X_test_category.head(5)
```

Out[313]:

	State_0	State_1
0	0.182063	0.817937
1	0.197828	0.802172
2	0.152964	0.847036
3	0.160920	0.839080
4	0.153846	0.846154

In [314]:
```
# roundoff all element in Dataframe | https://www.geeksforgeeks.org/python-pandas-dataframe-round/
X_train_category.round(2)
X_test_category.round(2)
X_test_category.head(5)
```

Out[314]:

	State_0	State_1
0	0.182063	0.817937
1	0.197828	0.802172
2	0.152964	0.847036
3	0.160920	0.839080
4	0.153846	0.846154

In [68]:
```
print(type(X_train_category))
print(type(X_test_category))
print(X_train_category.shape)
print(X_test_category.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
(33500, 2)
(16500, 2)
```

2.1.3 Make Data Model Ready: encoding clean_subcategories

In [315]:
```
# using value_count() to get each category's count, whene project is Approved
X_train_Clean_subcategory_One = X_train["clean_subcategories"][X_train['projec
t_is_approved']==1]
X_train_Clean_subcategory_One = X_train_Clean_subcategory_One.value_counts()
X_test_Clean_subcategory_One = X_test["clean_subcategories"][X_test['project_i
s_approved']==1]
X_test_Clean_subcategory_One = X_test_Clean_subcategory_One.value_counts()

# using value_count() to get each category's count, whene project is Not Appro
ved
X_train_Clean_subcategory_Zero = X_train["clean_subcategories"][X_train['proje
ct_is_approved']==0]
X_train_Clean_subcategory_Zero = X_train_Clean_subcategory_Zero.value_counts()
X_test_Clean_subcategory_Zero = X_test["clean_subcategories"][X_test['project_
is_approved']==0]
X_test_Clean_subcategory_Zero = X_test_Clean_subcategory_Zero.value_counts()

# using value_count() to get each category's count
X_train_Clean_subcategory_All = X_train["clean_subcategories"]
X_train_Clean_subcategory_All = X_train_Clean_subcategory_All.value_counts()
X_test_Clean_subcategory_All = X_test["clean_subcategories"]
X_test_Clean_subcategory_All = X_test_Clean_subcategory_All.value_counts()

# Creating a Dataframe, having response Table for Category, coulumn are derive
d above
X_train_response_tab_subcategory= pd.DataFrame(data=dict(X_train_Clean_subcate
gory_One=X_train_Clean_subcategory_One,X_train_Clean_subcategory_Zero=X_train_
Clean_subcategory_Zero,X_train_Clean_subcategory_All=X_train_Clean_subcategory
_All))
#X_test_response_tab_subcategory= pd.DataFrame(data=dict(X_test_Clean_subcateg
ory_One=X_test_Clean_subcategory_One,X_test_Clean_subcategory_Zero=X_test_Clea
n_subcategory_Zero,X_test_Clean_subcategory_All=X_test_Clean_subcategory_All))

# Adding two more coulmn, which are calculated as per the probablity i.e. Tota
l given category count per state/Total category count
X_train_response_tab_subcategory["State_0"]=X_train_response_tab_subcategory[
"X_train_Clean_subcategory_Zero"]/X_train_response_tab_subcategory["X_train_Cl
ean_subcategory_All"]
X_train_response_tab_subcategory["State_1"]=X_train_response_tab_subcategory[
"X_train_Clean_subcategory_One"]/X_train_response_tab_subcategory["X_train_Cle
an_subcategory_All"]
#X_test_response_tab_subcategory["State_0"]=X_test_response_tab_subcategory["X
_test_Clean_subcategory_Zero"]/X_test_response_tab_subcategory["X_test_Clean_s
ubcategory_All"]
#X_test_response_tab_subcategory["State_1"]=X_test_response_tab_subcategory["X
_test_Clean_subcategory_One"]/X_test_response_tab_subcategory["X_test_Clean_su
bcategory_All"]

# resetting an index,
X_train_response_tab_subcategory=X_train_response_tab_subcategory.reset_index
()
#X_test_response_tab_subcategory=X_test_response_tab_subcategory.reset_index()

# renaming an column header
X_train_response_tab_subcategory.columns=['clean_subcategories','X_train_Clean
_subcategory_One','X_train_Clean_subcategory_Zero','X_train_Clean_subcategory_
```

```
All','State_0','State_1']
print(X_train_response_tab_subcategory.shape)
X_train_response_tab_subcategory.head(5)

#X_test_response_tab_subcategory.columns=['clean_subcategories','X_test_Clean_
subcategory_One','X_test_Clean_subcategory_Zero','X_test_Clean_subcategory_Al
l','State_0','State_1']
#print(X_test_response_tab_subcategory.shape)
#X_test_response_tab_subcategory
```

```
(368, 6)
```

Out[315]:

	clean_subcategories	X_train_Clean_subcategory_One	X_train_Clean_subcategory_Ze
0	AppliedSciences	591.0	144.0
1	AppliedSciences CharacterEducation	8.0	3.0
2	AppliedSciences Civics_Government	3.0	1.0
3	AppliedSciences College_CareerPrep	104.0	22.0
4	AppliedSciences CommunityService	5.0	1.0

In [316]:
```
#(X_train_response_tab_subcategory.isna())
#X_train_response_tab_subcategory[:].isnull()
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Since thjis is the Response table, we find from Train data, and we need to u
se the same in Test data,
# we deleting rows whcih are nan
delete_index=[]
for i in range(X_train_response_tab_subcategory.shape[0]):
    if X_train_response_tab_subcategory['X_train_Clean_subcategory_One'].isnul
l().iloc[i]:
        delete_index.append(i)
for i in range(X_train_response_tab_subcategory.shape[0]):
    if X_train_response_tab_subcategory['X_train_Clean_subcategory_Zero'].isnu
ll().iloc[i]:
        delete_index.append(i)
print(delete_index)

X_train_response_tab_subcategory=X_train_response_tab_subcategory.dropna()
X_train_response_tab_subcategory.head(5)
```

```
[7, 19, 34, 60, 68, 80, 110, 122, 155, 168, 200, 237, 321, 323, 335, 338, 36
6, 25, 27, 29, 37, 42, 47, 59, 61, 62, 67, 74, 75, 76, 83, 94, 96, 99, 102, 1
03, 105, 109, 111, 112, 113, 117, 119, 120, 124, 129, 132, 139, 140, 144, 14
8, 149, 152, 157, 159, 161, 162, 163, 164, 165, 166, 167, 169, 171, 172, 173,
181, 184, 188, 192, 193, 194, 195, 202, 203, 204, 205, 206, 209, 210, 211, 21
2, 213, 216, 218, 220, 221, 223, 227, 228, 230, 233, 235, 242, 253, 256, 259,
261, 263, 276, 281, 283, 299, 302, 310, 313, 324, 328, 331, 332, 339, 341, 34
3, 345, 348, 350]
```

Out[316]:

	clean_subcategories	X_train_Clean_subcategory_One	X_train_Clean_subcategory_Ze
0	AppliedSciences	591.0	144.0
1	AppliedSciences CharacterEducation	8.0	3.0
2	AppliedSciences Civics_Government	3.0	1.0
3	AppliedSciences College_CareerPrep	104.0	22.0
4	AppliedSciences CommunityService	5.0	1.0

In [317]:
```
# create a new Dataframe for X_train_subcategory
X_train_subcategory=pd.DataFrame(data=dict(clean_subcategories=X_train["clean_
subcategories"],project_is_approved=X_train["project_is_approved"]))
X_train_subcategory

# create a new Dataframe for X_test_subcategory
X_test_subcategory=pd.DataFrame(data=dict(clean_subcategories=X_test["clean_su
bcategories"],project_is_approved=X_test["project_is_approved"]))
X_test_subcategory.head(5)
```

Out[317]:

	clean_subcategories	project_is_approved
31254	AppliedSciences	1
35307	SpecialNeeds	1
23220	Gym_Fitness	1
34585	AppliedSciences College_CareerPrep	1
46330	EarlyDevelopment Literature_Writing	1

In [318]:
```
# join X_train_subcategory with response table we created earlier
X_train_subcategory=pd.merge(X_train_subcategory,X_train_response_tab_subcateg
ory, on='clean_subcategories',how='left')
print(X_train_subcategory.shape)
X_train_subcategory

# join X_test_subcategory with response table we created earlier
# NOTE: we are tallying x_test with response table whihc we get from x_train
X_test_subcategory=pd.merge(X_test_subcategory,X_train_response_tab_subcategor
y, on='clean_subcategories',how='left')
print(X_test_subcategory.shape)
X_test_subcategory.head(5)
```

```
(33500, 7)
(16500, 7)
```

Out[318]:

	clean_subcategories	project_is_approved	X_train_Clean_subcategory_One	X_train_C
0	AppliedSciences	1	591.0	144.0
1	SpecialNeeds	1	1034.0	255.0
2	Gym_Fitness	1	304.0	67.0
3	AppliedSciences College_CareerPrep	1	104.0	22.0
4	EarlyDevelopment Literature_Writing	1	60.0	12.0

In [319]:
```
# dropping unwanted column names
X_train_subcategory.drop(['clean_subcategories','project_is_approved','X_train
_Clean_subcategory_One','X_train_Clean_subcategory_Zero','X_train_Clean_subcat
egory_All'], axis=1, inplace=True)
print(X_train_subcategory.shape)
X_train_subcategory

# dropping unwanted column names
X_test_subcategory.drop(['clean_subcategories','project_is_approved','X_train_
Clean_subcategory_One','X_train_Clean_subcategory_Zero','X_train_Clean_subcate
gory_All'], axis=1, inplace=True)
print(X_test_subcategory.shape)
X_test_subcategory.head(5)
```

```
(33500, 2)
(16500, 2)
```

Out[319]:

	State_0	State_1
0	0.195918	0.804082
1	0.197828	0.802172
2	0.180593	0.819407
3	0.174603	0.825397
4	0.166667	0.833333

In [292]:
```
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Checking if there is nay nan in Test dataframe
nan_index_tr=[]
for i in range(X_train_subcategory.shape[0]):
    if X_train_subcategory['State_0'].isnull().iloc[i]:
        nan_index_tr.append(i)
for i in range(X_train_subcategory.shape[0]):
    if X_train_subcategory['State_1'].isnull().iloc[i]:
        nan_index_tr.append(i)
#print(nan_index_tr)

nan_index_te=[]
for i in range(X_test_subcategory.shape[0]):
    if X_test_subcategory['State_0'].isnull().iloc[i]:
        nan_index_te.append(i)
for i in range(X_test_subcategory.shape[0]):
    if X_test_subcategory['State_1'].isnull().iloc[i]:
        nan_index_te.append(i)
#print(nan_index_te)
```

In [320]:
```
# For a category which is not there in train data and present in test data, we will
# encode them with default values Ex: in our test data if have State: D then we encode it as [0.5, 0.5]
X_train_subcategory.iloc[nan_index_tr,[0,1]]=[0.5, 0.5]
X_train_subcategory.iloc[nan_index_tr,[0,1]]

X_test_subcategory.iloc[nan_index_te,[0,1]]=[0.5, 0.5]
X_test_subcategory.iloc[nan_index_te,[0,1]]
X_test_subcategory.head(5)
```

Out[320]:

	State_0	State_1
0	0.195918	0.804082
1	0.197828	0.802172
2	0.180593	0.819407
3	0.174603	0.825397
4	0.166667	0.833333

In [321]:
```
# roundoff all element in Dataframe | https://www.geeksforgeeks.org/python-pandas-dataframe-round/
X_train_subcategory.round(2)
X_test_subcategory.round(2)
X_test_subcategory.head(5)
```

Out[321]:

	State_0	State_1
0	0.195918	0.804082
1	0.197828	0.802172
2	0.180593	0.819407
3	0.174603	0.825397
4	0.166667	0.833333

In [77]:
```
print(type(X_train_subcategory))
print(type(X_test_subcategory))
print(X_train_subcategory.shape)
print(X_test_subcategory.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
(33500, 2)
(16500, 2)
```

2.1.4 Make Data Model Ready: encoding project_grade_categor

In [78]:
```
# using value_count() to get each category's count, whene project is Approved
X_train_Clean_grade_One = X_train["clean_grade"][X_train['project_is_approved'
]==1]
X_train_Clean_grade_One = X_train_Clean_grade_One.value_counts()
X_test_Clean_grade_One = X_test["clean_grade"][X_test['project_is_approved']==
1]
X_test_Clean_grade_One = X_test_Clean_grade_One.value_counts()

# using value_count() to get each category's count, whene project is Not Appro
ved
X_train_Clean_grade_Zero = X_train["clean_grade"][X_train['project_is_approve
d']==0]
X_train_Clean_grade_Zero = X_train_Clean_grade_Zero.value_counts()
X_test_Clean_grade_Zero = X_test["clean_grade"][X_test['project_is_approved']=
=0]
X_test_Clean_grade_Zero = X_test_Clean_grade_Zero.value_counts()

# using value_count() to get each category's count
X_train_Clean_grade_All = X_train["clean_grade"]
X_train_Clean_grade_All = X_train_Clean_grade_All.value_counts()
X_test_Clean_grade_All = X_test["clean_grade"]
X_test_Clean_grade_All = X_test_Clean_grade_All.value_counts()

# Creating a Dataframe, having response Table for Category, coulumn are derive
d above
X_train_response_tab_grade= pd.DataFrame(data=dict(X_train_Clean_grade_One=X_t
rain_Clean_grade_One,X_train_Clean_grade_Zero=X_train_Clean_grade_Zero,X_train
_Clean_grade_All=X_train_Clean_grade_All))
#X_test_response_tab_grade= pd.DataFrame(data=dict(X_test_Clean_grade_One=X_te
st_Clean_grade_One,X_test_Clean_grade_Zero=X_test_Clean_grade_Zero,X_test_Clea
n_grade_All=X_test_Clean_grade_All))

# Adding two more coulmn, which are calculated as per the probablity i.e. Tota
l given category count per state/Total category count
X_train_response_tab_grade["State_0"]=X_train_response_tab_grade["X_train_Clea
n_grade_Zero"]/X_train_response_tab_grade["X_train_Clean_grade_All"]
X_train_response_tab_grade["State_1"]=X_train_response_tab_grade["X_train_Clea
n_grade_One"]/X_train_response_tab_grade["X_train_Clean_grade_All"]
#X_test_response_tab_grade["State_0"]=X_test_response_tab_grade["X_test_Clean_
grade_Zero"]/X_test_response_tab_grade["X_test_Clean_grade_All"]
#X_test_response_tab_grade["State_1"]=X_test_response_tab_grade["X_test_Clean_
grade_One"]/X_test_response_tab_grade["X_test_Clean_grade_All"]

# resetting an index,
X_train_response_tab_grade=X_train_response_tab_grade.reset_index()
#X_test_response_tab_grade=X_test_response_tab_grade.reset_index()

# renaming an column header
X_train_response_tab_grade.columns=['clean_grade','X_train_Clean_grade_One','X
_train_Clean_grade_Zero','X_train_Clean_grade_All','State_0','State_1']
print(X_train_response_tab_grade.shape)
X_train_response_tab_grade

#X_test_response_tab_grade.columns=['clean_grade','X_test_Clean_grade_One','X_
test_Clean_grade_Zero','X_test_Clean_grade_All','State_0','State_1']
```

```
#print(X_test_response_tab_grade.shape)
#X_test_response_tab_grade
```

```
(4, 6)
```

Out[78]:

	clean_grade	X_train_Clean_grade_One	X_train_Clean_grade_Zero	X_train_Clean_gra
0	PreK-2	11582	2058	13640
1	3-5	9595	1714	11309
2	6-8	4378	840	5218
3	9-12	2777	556	3333

In [79]:

```
#(X_train_response_tab_grade.isna())
#X_train_response_tab_grade[:].isnull()
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Since thjis is the Response table, we find from Train data, and we need to u
se the same in Test data,
# we deleting rows whcih are nan
delete_index=[]
for i in range(X_train_response_tab_grade.shape[0]):
    if X_train_response_tab_grade['X_train_Clean_grade_One'].isnull().iloc[i]:
        delete_index.append(i)
for i in range(X_train_response_tab_grade.shape[0]):
    if X_train_response_tab_grade['X_train_Clean_grade_Zero'].isnull().iloc[i
]:
        delete_index.append(i)
print(delete_index)

#X_train_response_tab_grade=X_train_response_tab_grade.dropna()
#X_train_response_tab_grade#
```

```
[]
```

In [322]:
```
# create a new Dataframe for X_train_grade
X_train_grade=pd.DataFrame(data=dict(clean_grade=X_train["clean_grade"],projec
t_is_approved=X_train["project_is_approved"]))
X_train_grade

# create a new Dataframe for X_test_grade
X_test_grade=pd.DataFrame(data=dict(clean_grade=X_test["clean_grade"],project_
is_approved=X_test["project_is_approved"]))
X_test_grade.head(5)
```

Out[322]:

	clean_grade	project_is_approved
31254	6-8	1
35307	PreK-2	1
23220	PreK-2	1
34585	3-5	1
46330	PreK-2	1

In [323]:
```
# join X_train_grade with response table we created earlier
X_train_grade=pd.merge(X_train_grade,X_train_response_tab_grade, on='clean_gra
de',how='left')
print(X_train_grade.shape)
X_train_grade

# join X_test_grade with response table we created earlier
# NOTE: we are tallying x_test with response table whihc we get from x_train
X_test_grade=pd.merge(X_test_grade,X_train_response_tab_grade, on='clean_grad
e',how='left')
print(X_test_grade.shape)
X_test_grade.head(5)
```

```
(33500, 7)
(16500, 7)
```

Out[323]:

	clean_grade	project_is_approved	X_train_Clean_grade_One	X_train_Clean_grade_Ze
0	6-8	1	4378	840
1	PreK-2	1	11582	2058
2	PreK-2	1	11582	2058
3	3-5	1	9595	1714
4	PreK-2	1	11582	2058

In [324]:
```
# dropping unwanted column names
X_train_grade.drop(['clean_grade','project_is_approved','X_train_Clean_grade_One','X_train_Clean_grade_Zero','X_train_Clean_grade_All'], axis=1, inplace=True)
print(X_train_grade.shape)
X_train_grade

# dropping unwanted column names
X_test_grade.drop(['clean_grade','project_is_approved','X_train_Clean_grade_One','X_train_Clean_grade_Zero','X_train_Clean_grade_All'], axis=1, inplace=True)
print(X_test_grade.shape)
X_test_grade.head(5)
```

```
(33500, 2)
(16500, 2)
```

Out[324]:

	State_0	State_1
0	0.160981	0.839019
1	0.150880	0.849120
2	0.150880	0.849120
3	0.151561	0.848439
4	0.150880	0.849120

In [83]:
```
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with-nan-in-pandas-dataframe
# Checking if there is nay nan in Test dataframe
nan_index_tr=[]
for i in range(X_train_grade.shape[0]):
    if X_train_grade['State_0'].isnull().iloc[i]:
        nan_index_tr.append(i)
for i in range(X_train_grade.shape[0]):
    if X_train_grade['State_1'].isnull().iloc[i]:
        nan_index_tr.append(i)
print(nan_index_tr)

nan_index_te=[]
for i in range(X_test_grade.shape[0]):
    if X_test_grade['State_0'].isnull().iloc[i]:
        nan_index_te.append(i)
for i in range(X_test_grade.shape[0]):
    if X_test_grade['State_1'].isnull().iloc[i]:
        nan_index_te.append(i)
print(nan_index_te)
```

```
[]
[]
```

In [325]:
```
# roundoff all element in Dataframe | https://www.geeksforgeeks.org/python-pandas-dataframe-round/
X_train_grade.round(2)
X_test_grade.round(2)
X_test_grade.head(5)
```

Out[325]:

	State_0	State_1
0	0.160981	0.839019
1	0.150880	0.849120
2	0.150880	0.849120
3	0.151561	0.848439
4	0.150880	0.849120

In [84]:
```
print(type(X_train_grade))
print(type(X_test_grade))
print(X_train_grade.shape)
print(X_test_grade.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
(33500, 2)
(16500, 2)
```

2.1.5 Make Data Model Ready: encoding teacher_prefix

In [85]:
```
# using value_count() to get each category's count, whene project is Approved
X_train_tea_pfx_One = X_train["clean_tea_pfx"][X_train['project_is_approved']==1]
X_train_tea_pfx_One = X_train_tea_pfx_One.value_counts()
X_test_tea_pfx_One = X_test["clean_tea_pfx"][X_test['project_is_approved']==1]
X_test_tea_pfx_One = X_test_tea_pfx_One.value_counts()

# using value_count() to get each category's count, whene project is Not Approved
X_train_tea_pfx_Zero = X_train["clean_tea_pfx"][X_train['project_is_approved']==0]
X_train_tea_pfx_Zero = X_train_tea_pfx_Zero.value_counts()
X_test_tea_pfx_Zero = X_test["clean_tea_pfx"][X_test['project_is_approved']==0]
X_test_tea_pfx_Zero = X_test_tea_pfx_Zero.value_counts()

# using value_count() to get each category's count
X_train_tea_pfx_All = X_train["clean_tea_pfx"]
X_train_tea_pfx_All = X_train_tea_pfx_All.value_counts()
X_test_tea_pfx_All = X_test["clean_tea_pfx"]
X_test_tea_pfx_All = X_test_tea_pfx_All.value_counts()

# Creating a Dataframe, having response Table for Category, coulumn are derived above
X_train_response_tab_tea_pfx= pd.DataFrame(data=dict(X_train_tea_pfx_One=X_train_tea_pfx_One,X_train_tea_pfx_Zero=X_train_tea_pfx_Zero,X_train_tea_pfx_All=X_train_tea_pfx_All))
#X_test_response_tab_tea_pfx= pd.DataFrame(data=dict(X_test_tea_pfx_One=X_test_tea_pfx_One,X_test_tea_pfx_Zero=X_test_tea_pfx_Zero,X_test_tea_pfx_All=X_test_tea_pfx_All))

# Adding two more coulmn, which are calculated as per the probablity i.e. Total given category count per state/Total category count
X_train_response_tab_tea_pfx["State_0"]=X_train_response_tab_tea_pfx["X_train_tea_pfx_Zero"]/X_train_response_tab_tea_pfx["X_train_tea_pfx_All"]
X_train_response_tab_tea_pfx["State_1"]=X_train_response_tab_tea_pfx["X_train_tea_pfx_One"]/X_train_response_tab_tea_pfx["X_train_tea_pfx_All"]
#X_test_response_tab_tea_pfx["State_0"]=X_test_response_tab_tea_pfx["X_test_tea_pfx_Zero"]/X_test_response_tab_tea_pfx["X_test_tea_pfx_All"]
#X_test_response_tab_tea_pfx["State_1"]=X_test_response_tab_tea_pfx["X_test_tea_pfx_One"]/X_test_response_tab_tea_pfx["X_test_tea_pfx_All"]

# resetting an index,
X_train_response_tab_tea_pfx=X_train_response_tab_tea_pfx.reset_index()
#X_test_response_tab_tea_pfx=X_test_response_tab_tea_pfx.reset_index()

# renaming an column header
X_train_response_tab_tea_pfx.columns=['clean_tea_pfx','X_train_tea_pfx_One','X_train_tea_pfx_Zero','X_train_tea_pfx_All','State_0','State_1']
print(X_train_response_tab_tea_pfx.shape)
X_train_response_tab_tea_pfx

#X_test_response_tab_tea_pfx.columns=['clean_tea_pfx','X_test_tea_pfx_One','X_test_tea_pfx_Zero','X_test_tea_pfx_All','State_0','State_1']
#print(X_test_response_tab_tea_pfx.shape)
#X_test_response_tab_tea_pfx
```

```
(6, 6)
```

Out[85]:

	clean_tea_pfx	X_train_tea_pfx_One	X_train_tea_pfx_Zero	X_train_tea_pfx_All	State
0		2	NaN	2	NaN
1	Dr	1	1.0	2	0.5000
2	Mr	2719	531.0	3250	0.1633
3	Mrs	14889	2616.0	17505	0.1494
4	Ms	10158	1863.0	12021	0.1549
5	Teacher	563	157.0	720	0.2180

In [86]:
```
#(X_train_response_tab_tea_pfx.isna())
#X_train_response_tab_tea_pfx[:].isnull()
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Since thjis is the Response table, we find from Train data, and we need to u
se the same in Test data,
# we deleting rows whcih are nan
delete_index=[]
for i in range(X_train_response_tab_tea_pfx.shape[0]):
    if X_train_response_tab_tea_pfx['X_train_tea_pfx_One'].isnull().iloc[i]:
        delete_index.append(i)
for i in range(X_train_response_tab_tea_pfx.shape[0]):
    if X_train_response_tab_tea_pfx['X_train_tea_pfx_Zero'].isnull().iloc[i]:
        delete_index.append(i)
print(delete_index)

X_train_response_tab_tea_pfx=X_train_response_tab_tea_pfx.dropna()
X_train_response_tab_tea_pfx
```

```
[0]
```

Out[86]:

	clean_tea_pfx	X_train_tea_pfx_One	X_train_tea_pfx_Zero	X_train_tea_pfx_All	State
1	Dr	1	1.0	2	0.5000
2	Mr	2719	531.0	3250	0.1633
3	Mrs	14889	2616.0	17505	0.1494
4	Ms	10158	1863.0	12021	0.1549
5	Teacher	563	157.0	720	0.2180

In [326]:
```
# create a new Dataframe for X_train_tea_pfx
X_train_tea_pfx=pd.DataFrame(data=dict(clean_tea_pfx=X_train["clean_tea_pfx"],
project_is_approved=X_train["project_is_approved"]))
X_train_tea_pfx

# create a new Dataframe for X_test_tea_pfx
X_test_tea_pfx=pd.DataFrame(data=dict(clean_tea_pfx=X_test["clean_tea_pfx"],pr
oject_is_approved=X_test["project_is_approved"]))
X_test_tea_pfx.head(5)
```

Out[326]:

	clean_tea_pfx	project_is_approved
31254	Mrs	1
35307	Ms	1
23220	Mrs	1
34585	Mrs	1
46330	Mrs	1

In [327]:
```
# join X_train_tea_pfx with response table we created earlier
X_train_tea_pfx=pd.merge(X_train_tea_pfx,X_train_response_tab_tea_pfx, on='cle
an_tea_pfx',how='left')
print(X_train_tea_pfx.shape)
X_train_tea_pfx

# join X_test_tea_pfx with response table we created earlier
# NOTE: we are tallying x_test with response table whihc we get from x_train
X_test_tea_pfx=pd.merge(X_test_tea_pfx,X_train_response_tab_tea_pfx, on='clean
_tea_pfx',how='left')
print(X_test_tea_pfx.shape)
X_test_tea_pfx.head(5)
```

```
(33500, 7)
(16500, 7)
```

Out[327]:

	clean_tea_pfx	project_is_approved	X_train_tea_pfx_One	X_train_tea_pfx_Zero	X_tra
0	Mrs	1	14889	2616.0	1750
1	Ms	1	10158	1863.0	1202
2	Mrs	1	14889	2616.0	1750
3	Mrs	1	14889	2616.0	1750
4	Mrs	1	14889	2616.0	1750

In [328]:
```
# dropping unwanted column names
X_train_tea_pfx.drop(['clean_tea_pfx','project_is_approved','X_train_tea_pfx_O
ne','X_train_tea_pfx_Zero','X_train_tea_pfx_All'], axis=1, inplace=True)
print(X_train_tea_pfx.shape)
X_train_tea_pfx

# dropping unwanted column names
X_test_tea_pfx.drop(['clean_tea_pfx','project_is_approved','X_train_tea_pfx_On
e','X_train_tea_pfx_Zero','X_train_tea_pfx_All'], axis=1, inplace=True)
print(X_test_tea_pfx.shape)
X_test_tea_pfx.head(5)
```

```
(33500, 2)
(16500, 2)
```

Out[328]:

	State_0	State_1
0	0.149443	0.850557
1	0.154979	0.845021
2	0.149443	0.850557
3	0.149443	0.850557
4	0.149443	0.850557

In [90]:
```
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Checking if there is nay nan in Test dataframe
nan_index_tr=[]
for i in range(X_train_tea_pfx.shape[0]):
    if X_train_tea_pfx['State_0'].isnull().iloc[i]:
        nan_index_tr.append(i)
for i in range(X_train_tea_pfx.shape[0]):
    if X_train_tea_pfx['State_1'].isnull().iloc[i]:
        nan_index_tr.append(i)
print(nan_index_tr)

nan_index_te=[]
for i in range(X_test_tea_pfx.shape[0]):
    if X_test_tea_pfx['State_0'].isnull().iloc[i]:
        nan_index_te.append(i)
for i in range(X_test_tea_pfx.shape[0]):
    if X_test_tea_pfx['State_1'].isnull().iloc[i]:
        nan_index_te.append(i)
print(nan_index_te)
```

```
[6867, 19207, 6867, 19207]
[]
```

In [91]:
```
# For a category which is not there in train data and present in test data, we will
# encode them with default values Ex: in our test data if have State: D then we encode it as [0.5, 0.5]
X_train_tea_pfx.iloc[nan_index_tr,[0,1]]=[0.5, 0.5]
X_train_tea_pfx.iloc[nan_index_tr,[0,1]]

X_test_tea_pfx.iloc[nan_index_te,[0,1]]=[0.5, 0.5]
X_test_tea_pfx.iloc[nan_index_te,[0,1]]
```

Out[91]:

	State_0	State_1

In [94]:
```
print(type(X_train_tea_pfx))
print(type(X_test_tea_pfx))
print(X_train_tea_pfx.shape)
print(X_test_tea_pfx.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
(33500, 2)
(16500, 2)
```

In [329]:
```
# roundoff all element in Dataframe | https://www.geeksforgeeks.org/python-pandas-dataframe-round/
X_train_tea_pfx.round(2)
X_test_tea_pfx.round(2)
X_test_tea_pfx.head(5)
```

Out[329]:

	State_0	State_1
0	0.149443	0.850557
1	0.154979	0.845021
2	0.149443	0.850557
3	0.149443	0.850557
4	0.149443	0.850557

2.1.6 Make Data Model Ready: encoding project_resource_summary

In [330]:
```
# using value_count() to get each category's count, whene project is Approved
X_train_prj_res_sum_One = X_train["project_resource_summary"][X_train['project
_is_approved']==1]
X_train_prj_res_sum_One = X_train_prj_res_sum_One.value_counts()
X_test_prj_res_sum_One = X_test["project_resource_summary"][X_test['project_is
_approved']==1]
X_test_prj_res_sum_One = X_test_prj_res_sum_One.value_counts()

# using value_count() to get each category's count, whene project is Not Appro
ved
X_train_prj_res_sum_Zero = X_train["project_resource_summary"][X_train['projec
t_is_approved']==0]
X_train_prj_res_sum_Zero = X_train_prj_res_sum_Zero.value_counts()
X_test_prj_res_sum_Zero = X_test["project_resource_summary"][X_test['project_i
s_approved']==0]
X_test_prj_res_sum_Zero = X_test_prj_res_sum_Zero.value_counts()

# using value_count() to get each category's count
X_train_prj_res_sum_All = X_train["project_resource_summary"]
X_train_prj_res_sum_All = X_train_prj_res_sum_All.value_counts()
X_test_prj_res_sum_All = X_test["project_resource_summary"]
X_test_prj_res_sum_All = X_test_prj_res_sum_All.value_counts()

# Creating a Dataframe, having response Table for Category, coulumn are derive
d above
X_train_response_tab_prj_res_sum= pd.DataFrame(data=dict(X_train_prj_res_sum_O
ne=X_train_prj_res_sum_One,X_train_prj_res_sum_Zero=X_train_prj_res_sum_Zero,X
_train_prj_res_sum_All=X_train_prj_res_sum_All))
#X_test_response_tab_prj_res_sum= pd.DataFrame(data=dict(X_test_prj_res_sum_On
e=X_test_prj_res_sum_One,X_test_prj_res_sum_Zero=X_test_prj_res_sum_Zero,X_tes
t_prj_res_sum_All=X_test_prj_res_sum_All))

# Adding two more coulmn, which are calculated as per the probablity i.e. Tota
l given category count per state/Total category count
X_train_response_tab_prj_res_sum["State_0"]=X_train_response_tab_prj_res_sum[
"X_train_prj_res_sum_Zero"]/X_train_response_tab_prj_res_sum["X_train_prj_res_
sum_All"]
X_train_response_tab_prj_res_sum["State_1"]=X_train_response_tab_prj_res_sum[
"X_train_prj_res_sum_One"]/X_train_response_tab_prj_res_sum["X_train_prj_res_s
um_All"]
#X_test_response_tab_prj_res_sum["State_0"]=X_test_response_tab_prj_res_sum["X
_test_prj_res_sum_Zero"]/X_test_response_tab_prj_res_sum["X_test_prj_res_sum_A
ll"]
#X_test_response_tab_prj_res_sum["State_1"]=X_test_response_tab_prj_res_sum["X
_test_prj_res_sum_One"]/X_test_response_tab_prj_res_sum["X_test_prj_res_sum_Al
l"]

# resetting an index,
X_train_response_tab_prj_res_sum=X_train_response_tab_prj_res_sum.reset_index
()
#X_test_response_tab_prj_res_sum=X_test_response_tab_prj_res_sum.reset_index()

# renaming an column header
X_train_response_tab_prj_res_sum.columns=['project_resource_summary','X_train_
prj_res_sum_One','X_train_prj_res_sum_Zero','X_train_prj_res_sum_All','State_
0','State_1']
```

```
print(X_train_response_tab_prj_res_sum.shape)
X_train_response_tab_prj_res_sum.head(5)

#X_test_response_tab_prj_res_sum.columns=['project_resource_summary','X_test_p
rj_res_sum_One','X_test_prj_res_sum_Zero','X_test_prj_res_sum_All','State_
0','State_1']
#print(X_test_response_tab_prj_res_sum.shape)
#X_test_response_tab_prj_res_sum
```

```
(33382, 6)
```

Out[330]:

	project_resource_summary	X_train_prj_res_sum_One	X_train_prj_res_sum_Zero	X_t
0	*My students need 30 books to improve literacy...	1.0	NaN	1
1	-My students need mentor texts that will showc...	1.0	NaN	1
2	Do you remember sitting by the campfire and te...	1.0	NaN	1
3	I need a dash cam to protect myself and studen...	1.0	NaN	1
4	My school is a public, Title 1 elementary scho...	1.0	NaN	1

In [293]:

```
#(X_train_response_tab_prj_res_sum.isna())
#X_train_response_tab_prj_res_sum[:].isnull()
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Since thjis is the Response table, we find from Train data, and we need to u
se the same in Test data,
# we deleting rows whcih are nan
delete_index=[]
for i in range(X_train_response_tab_prj_res_sum.shape[0]):
    if X_train_response_tab_prj_res_sum['X_train_prj_res_sum_One'].isnull().il
oc[i]:
        delete_index.append(i)
for i in range(X_train_response_tab_prj_res_sum.shape[0]):
    if X_train_response_tab_prj_res_sum['X_train_prj_res_sum_Zero'].isnull().i
loc[i]:
        delete_index.append(i)
#print(delete_index)

X_train_response_tab_prj_res_sum=X_train_response_tab_prj_res_sum.dropna()
#X_train_response_tab_prj_res_sum
```

In [331]:
```
# create a new Dataframe for X_train_prj_res_sum
X_train_prj_res_sum=pd.DataFrame(data=dict(project_resource_summary=X_train["p
roject_resource_summary"],project_is_approved=X_train["project_is_approved"]))
X_train_prj_res_sum

# create a new Dataframe for X_test_prj_res_sum
X_test_prj_res_sum=pd.DataFrame(data=dict(project_resource_summary=X_test["pro
ject_resource_summary"],project_is_approved=X_test["project_is_approved"]))
X_test_prj_res_sum.head(5)
```

Out[331]:

	project_resource_summary	project_is_approved
31254	My students need scooter boards to investigate...	1
35307	My students need books that will support healt...	1
23220	My students need gross motor equipment for use...	1
34585	My students need 6 HP Chromebooks for use in t...	1
46330	My students need tables and chairs that they c...	1

In [332]:
```
# join X_train_prj_res_sum with response table we created earlier
X_train_prj_res_sum=pd.merge(X_train_prj_res_sum,X_train_response_tab_prj_res_
sum, on='project_resource_summary',how='left')
print(X_train_prj_res_sum.shape)
X_train_prj_res_sum

# join X_test_prj_res_sum with response table we created earlier
# NOTE: we are tallying x_test with response table whihc we get from x_train
X_test_prj_res_sum=pd.merge(X_test_prj_res_sum,X_train_response_tab_prj_res_su
m, on='project_resource_summary',how='left')
print(X_test_prj_res_sum.shape)
X_test_prj_res_sum.head(5)
```

```
(33500, 7)
(16500, 7)
```

Out[332]:

	project_resource_summary	project_is_approved	X_train_prj_res_sum_One	X_train_p
0	My students need scooter boards to investigate...	1	NaN	NaN
1	My students need books that will support healt...	1	NaN	NaN
2	My students need gross motor equipment for use...	1	NaN	NaN
3	My students need 6 HP Chromebooks for use in t...	1	NaN	NaN
4	My students need tables and chairs that they c...	1	NaN	NaN

In [333]:
```
# dropping unwanted column names
X_train_prj_res_sum.drop(['project_resource_summary','project_is_approved','X_
train_prj_res_sum_One','X_train_prj_res_sum_Zero','X_train_prj_res_sum_All'],
axis=1, inplace=True)
print(X_train_prj_res_sum.shape)
X_train_prj_res_sum

# dropping unwanted column names
X_test_prj_res_sum.drop(['project_resource_summary','project_is_approved','X_t
rain_prj_res_sum_One','X_train_prj_res_sum_Zero','X_train_prj_res_sum_All'], a
xis=1, inplace=True)
print(X_test_prj_res_sum.shape)
X_test_prj_res_sum.head(5)
```

```
(33500, 2)
(16500, 2)
```

Out[333]:

	State_0	State_1
0	NaN	NaN
1	NaN	NaN
2	NaN	NaN
3	NaN	NaN
4	NaN	NaN

In [294]:
```
# https://stackoverflow.com/questions/14016247/find-integer-index-of-rows-with
-nan-in-pandas-dataframe
# Checking if there is nay nan in Test dataframe
nan_index_tr=[]
for i in range(X_train_prj_res_sum.shape[0]):
    if X_train_prj_res_sum['State_0'].isnull().iloc[i]:
        nan_index_tr.append(i)
for i in range(X_train_prj_res_sum.shape[0]):
    if X_train_prj_res_sum['State_1'].isnull().iloc[i]:
        nan_index_tr.append(i)
#print(nan_index_tr)

nan_index_te=[]
for i in range(X_test_prj_res_sum.shape[0]):
    if X_test_prj_res_sum['State_0'].isnull().iloc[i]:
        nan_index_te.append(i)
for i in range(X_test_prj_res_sum.shape[0]):
    if X_test_prj_res_sum['State_1'].isnull().iloc[i]:
        nan_index_te.append(i)
#print(nan_index_te)
```

In [334]:
```
# For a category which is not there in train data and present in test data, we will
# encode them with default values Ex: in our test data if have State: D then we encode it as [0.5, 0.5]
X_train_prj_res_sum.iloc[nan_index_tr,[0,1]]=[0.5, 0.5]
X_train_prj_res_sum.iloc[nan_index_tr,[0,1]]

X_test_prj_res_sum.iloc[nan_index_te,[0,1]]=[0.5, 0.5]
X_test_prj_res_sum.iloc[nan_index_te,[0,1]]
X_test_prj_res_sum.head(5)
```

Out[334]:

	State_0	State_1
0	NaN	NaN
1	NaN	NaN
2	NaN	NaN
3	NaN	NaN
4	NaN	NaN

In [335]:
```
# roundoff all element in Dataframe | https://www.geeksforgeeks.org/python-pandas-dataframe-round/
X_train_prj_res_sum.round(2)
X_test_prj_res_sum.round(2)
X_test_prj_res_sum.head(5)
```

Out[335]:

	State_0	State_1
0	NaN	NaN
1	NaN	NaN
2	NaN	NaN
3	NaN	NaN
4	NaN	NaN

In [106]:
```
print(type(X_train_prj_res_sum))
print(type(X_test_prj_res_sum))
print(X_train_prj_res_sum.shape)
print(X_test_prj_res_sum.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
(33500, 2)
(16500, 2)
```

2.2 Make Data Model Ready: encoding numerical, categorical features

In [107]:
```
# please write all the code with proper documentation, and proper titles for e
ach subsection
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debug
ging your code
# make sure you featurize train and test data separatly

# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the rea
der
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

2.2.1 Make Data Model Ready: encoding numerical | quantity

In [108]:
```
from sklearn.preprocessing import Normalizer
normalizer = Normalizer()
# normalizer.fit(X_train['price'].values)
# this will rise an error Expected 2D array, got 1D array instead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
normalizer.fit(X_train['quantity'].values.reshape(-1,1))

X_train_quantity_norm = normalizer.transform(X_train['quantity'].values.reshap
e(-1,1))
#X_cv_quantity_norm = normalizer.transform(X_cv['quantity'].values.reshape(-1,
1))
X_test_quantity_norm = normalizer.transform(X_test['quantity'].values.reshape(
-1,1))

print("quantity After vectorizations")
print(X_train_quantity_norm.shape, y_train.shape)
#print(X_cv_quantity_norm.shape, y_cv.shape)
print(X_test_quantity_norm.shape, y_test.shape)
print("="*100)
```

```
quantity After vectorizations
(33500, 1) (33500,)
(16500, 1) (16500,)
====================================================================================
======================
```

2.2.2 Make Data Model Ready: encoding numerical| teacher_number_of_previously_posted_projects

In [109]:
```
from sklearn.preprocessing import Normalizer
normalizer = Normalizer()
# normalizer.fit(X_train['price'].values)
# this will rise an error Expected 2D array, got 1D array instead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
normalizer.fit(X_train['teacher_number_of_previously_posted_projects'].values.
reshape(-1,1))

X_train_TprevPrj_norm = normalizer.transform(X_train['teacher_number_of_previo
usly_posted_projects'].values.reshape(-1,1))
#X_cv_TprevPrj_norm = normalizer.transform(X_cv['teacher_number_of_previously_
posted_projects'].values.reshape(-1,1))
X_test_TprevPrj_norm = normalizer.transform(X_test['teacher_number_of_previous
ly_posted_projects'].values.reshape(-1,1))

print("teacher_number_of_previously_posted_projects After vectorizations")
print(X_train_TprevPrj_norm.shape, y_train.shape)
#print(X_cv_TprevPrj_norm.shape, y_cv.shape)
print(X_test_TprevPrj_norm.shape, y_test.shape)
print("="*100)
```

```
teacher_number_of_previously_posted_projects After vectorizations
(33500, 1) (33500,)
(16500, 1) (16500,)
====================================================================================
======================
```

In [110]:
```
print(type(X_train_TprevPrj_norm))
print(type(X_test_TprevPrj_norm))
```

```
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
```

2.2.3 Make Data Model Ready: encoding numerical | price

In [111]:
```
from sklearn.preprocessing import Normalizer
normalizer = Normalizer()
# normalizer.fit(X_train['price'].values)
# this will rise an error Expected 2D array, got 1D array instead:
# array=[105.22 215.96  96.01 ... 368.98  80.53 709.67].
# Reshape your data either using
# array.reshape(-1, 1) if your data has a single feature
# array.reshape(1, -1)  if it contains a single sample.
normalizer.fit(X_train['price'].values.reshape(-1,1))

X_train_price_norm = normalizer.transform(X_train['price'].values.reshape(-1,1
))
#X_cv_price_norm = normalizer.transform(X_cv['price'].values.reshape(-1,1))
X_test_price_norm = normalizer.transform(X_test['price'].values.reshape(-1,1))

print("Price After vectorizations")
print(X_train_price_norm.shape, y_train.shape)
#print(X_cv_price_norm.shape, y_cv.shape)
print(X_test_price_norm.shape, y_test.shape)
print("="*100)
```

```
Price After vectorizations
(33500, 1) (33500,)
(16500, 1) (16500,)
====================================================================================
======================
```

2.3 Make Data Model Ready: encoding eassay, and project_title

In [112]:
```
# please write all the code with proper documentation, and proper titles for e
ach subsection
# go through documentations and blogs before you start coding
# first figure out what to do, and then think about how to do.
# reading and understanding error messages will be very much helpfull in debug
ging your code
# make sure you featurize train and test data separatly

# when you plot any graph make sure you use
    # a. Title, that describes your plot, this will be very helpful to the rea
der
    # b. Legends if needed
    # c. X-axis label
    # d. Y-axis label
```

2.3.1 Make Data Model Ready: project_essay | BOW

In [113]:
```
from sklearn.feature_extraction.text import CountVectorizer
# categorical, numerical features + project_title(BOW) + preprocessed_eassay
# (BOW with bi-grams with min_df=10 and max_features=5000)
vectorizer = CountVectorizer(min_df=10,ngram_range=(1,2), max_features=5000)
vectorizer.fit(X_train['essay'].values) # fit has to happen only on train data

# we use the fitted CountVectorizer to convert the text to vector
X_train_essay_bow = vectorizer.transform(X_train['essay'].values)
#X_cv_essay_bow = vectorizer.transform(X_cv['essay'].values)
X_test_essay_bow = vectorizer.transform(X_test['essay'].values)

print("Essay After vectorizations")
print(X_train_essay_bow.shape, y_train.shape)
#print(X_cv_essay_bow.shape, y_cv.shape)
print(X_test_essay_bow.shape, y_test.shape)
print("="*100)
```

```
Essay After vectorizations
(33500, 5000) (33500,)
(16500, 5000) (16500,)
====================================================================================================
```

2.3.2 Make Data Model Ready: project_title | BOW

In [114]:
```
vectorizer = CountVectorizer()
# categorical, numerical features + project_title(BOW) + preprocessed_eassay
# (BOW with bi-grams with min_df=10 and max_features=5000)
vectorizer = CountVectorizer(min_df=10,ngram_range=(1,2), max_features=5000)
vectorizer.fit(X_train['project_title'].values) # fit has to happen only on tr
ain data

# we use the fitted CountVectorizer to convert the text to vector
X_train_title_bow = vectorizer.transform(X_train['project_title'].values)
#X_cv_title_bow = vectorizer.transform(X_cv['project_title'].values)
X_test_title_bow = vectorizer.transform(X_test['project_title'].values)

print("project_title After vectorizations")
print(X_train_title_bow.shape, y_train.shape)
#print(X_cv_title_bow.shape, y_cv.shape)
print(X_test_title_bow.shape, y_test.shape)
#print(vectorizer.get_feature_names())
print("="*100)
```

```
project_title After vectorizations
(33500, 3405) (33500,)
(16500, 3405) (16500,)
====================================================================================================
```

2.3.3 Make Data Model Ready: project_essay | TFIDF

In [115]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
# categorical, numerical features + project_title(BOW) + preprocessed_eassay
# (TFIDF with bi-grams with min_df=10 and max_features=5000)
Tfidf_vectorizer = TfidfVectorizer(min_df=10,ngram_range=(1,2), max_features=5
000)

Tfidf_vectorizer.fit(X_train['essay'].values)

X_train_text_tfidf = Tfidf_vectorizer.transform(X_train['essay'].values)
#X_cv_text_tfidf = Tfidf_vectorizer.transform(X_cv['essay'].values)
X_test_text_tfidf = Tfidf_vectorizer.transform(X_test['essay'].values)

##print("Shape of matrix after one hot encodig ",text_tfidf.shape)

print("Essay After vectorizations")
print(X_train_text_tfidf.shape, y_train.shape)
#print(X_cv_text_tfidf.shape, y_cv.shape)
print(X_test_text_tfidf.shape, y_test.shape)
#print(Tfidf_vectorizer.get_feature_names())
print("="*100)
```

```
Essay After vectorizations
(33500, 5000) (33500,)
(16500, 5000) (16500,)
====================================================================================
======================
```

2.3.4 Make Data Model Ready: project_title | TFIDF

In [116]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
# categorical, numerical features + project_title(BOW) + preprocessed_eassay
# (TFIDF with bi-grams with min_df=10 and max_features=5000)
Tfidf_vectorizer = TfidfVectorizer(min_df=10,ngram_range=(1,2), max_features=5
000)

Tfidf_vectorizer.fit(X_train['project_title'].values)

X_train_title_tfidf = Tfidf_vectorizer.transform(X_train['project_title'].valu
es)
#X_cv_title_tfidf = Tfidf_vectorizer.transform(X_cv['project_title'].values)
X_test_title_tfidf = Tfidf_vectorizer.transform(X_test['project_title'].values
)

##print("Shape of matrix after one hot encodig ",text_tfidf.shape)

print("project_title After vectorizations")
print(X_train_title_tfidf.shape, y_train.shape)
#print(X_cv_title_tfidf.shape, y_cv.shape)
print(X_test_title_tfidf.shape, y_test.shape)
#print(Tfidf_vectorizer.get_feature_names())
print("="*100)
```

```
project_title After vectorizations
(33500, 3405) (33500,)
(16500, 3405) (16500,)
====================================================================================
======================
```

2.3.5 Make Data Model Ready: project_essay | AVG W2V

In [117]:
```
# average Word2Vec for Train Essay
# compute average word2vec for each review.
X_train_essay_avg_w2v = []; # the avg-w2v for each sentence/review is stored i
n this list
for sentence in tqdm(X_train['essay'].values): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    X_train_essay_avg_w2v.append(vector)

print(len(X_train_essay_avg_w2v))
print(len(X_train_essay_avg_w2v[0]))

# stronging variables into pickle files python: http://www.jessicayung.com/how
-to-use-pickle-to-save-and-load-variables-in-python/

import pickle
with open('X_train_essay_avg_w2v', 'wb') as f:
    pickle.dump(X_train_essay_avg_w2v, f)
```

```
100%|████████████████████████████████████████████████████████████████
██| 33500/33500 [00:13<00:00, 2400.62it/s]

33500
300
```

In [118]:
```
## stronging variables into pickle files python: http://www.jessicayung.com/ho
w-to-use-pickle-to-save-and-load-variables-in-python/
## make sure you have the glove_vectors file
#with open#('X_train_essay_avg_w2v', 'rb') as f:
#    X_train_essay_avg_w2v = pickle.load(f)
#
#print(len(X_train_essay_avg_w2v))
#print(len(X_train_essay_avg_w2v[0]))
```

In [119]:
```
# average Word2Vec for Test Essay
# compute average word2vec for each review.
X_test_essay_avg_w2v = []; # the avg-w2v for each sentence/review is stored in
this list
for sentence in tqdm(X_test['essay'].values): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    X_test_essay_avg_w2v.append(vector)

print(len(X_test_essay_avg_w2v))
print(len(X_test_essay_avg_w2v[0]))
```

```
100%|████████████████████████████████████████████████████████████████████
██| 16500/16500 [00:06<00:00, 2541.73it/s]

16500
300
```

2.3.6 Make Data Model Ready: project_title | AVG W2V

In [120]:
```
# average Word2Vec for Train Title
# compute average word2vec for each review.
X_train_title_avg_w2v = []; # the avg-w2v for each sentence/review is stored i
n this list
for sentence in tqdm(X_train['project_title'].values): # for each review/sente
nce
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    X_train_title_avg_w2v.append(vector)

print(len(X_train_title_avg_w2v))
print(len(X_train_title_avg_w2v[0]))
```

```
100%|████████████████████████████████████████████████████████████████████
| 33500/33500 [00:00<00:00, 126752.60it/s]

33500
300
```

In [121]:
```
# average Word2Vec for Test Essay
# compute average word2vec for each review.
X_test_title_avg_w2v = []; # the avg-w2v for each sentence/review is stored in
this list
for sentence in tqdm(X_test['project_title'].values): # for each review/senten
ce
    vector = np.zeros(300) # as word vectors are of zero length
    cnt_words =0; # num of words with a valid vector in the sentence/review
    for word in sentence.split(): # for each word in a review/sentence
        if word in glove_words:
            vector += model[word]
            cnt_words += 1
    if cnt_words != 0:
        vector /= cnt_words
    X_test_title_avg_w2v.append(vector)

print(len(X_test_title_avg_w2v))
print(len(X_test_title_avg_w2v[0]))
```

```
100%|████████████████████████████████████████████████████████████████████
| 16500/16500 [00:00<00:00, 130268.92it/s]

16500
300
```

2.3.7 Make Data Model Ready: project_essay | TFIDF W2V

In [122]:
```
# S = ["abc def pqr", "def def def abc", "pqr pqr def"]
Tr_tfidf_model_essay = TfidfVectorizer()
Tr_tfidf_model_essay.fit(X_train['essay'].values)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(Tr_tfidf_model_essay.get_feature_names(), list(Tr_tfidf_
model_essay.idf_)))
tr_essay_tfidf_words = set(Tr_tfidf_model_essay.get_feature_names())
```

In [123]:
```
# TFIDF weighted Word2Vec for train essay
# compute average word2vec for each review.
tr_tfidf_w2v_essay_vectors = []; # the avg-w2v for each sentence/review is sto
red in this list
for sentence in tqdm(X_train['essay'].values): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/revie
w
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in tr_essay_tfidf_words):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf v
alue((sentence.count(word)/len(sentence.split())))
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split
())) # getting the tfidf value for each word
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    tr_tfidf_w2v_essay_vectors.append(vector)

print(len(tr_tfidf_w2v_essay_vectors))
print(len(tr_tfidf_w2v_essay_vectors[0]))
```

```
100%|████████████████████████████████████████████████████████████████
██| 33500/33500 [6:51:27<00:00,  1.36it/s]

33500
300
```

In [124]:
```
# S = ["abc def pqr", "def def def abc", "pqr pqr def"]
te_tfidf_model_essay = TfidfVectorizer()
te_tfidf_model_essay.fit(X_test['essay'].values)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(te_tfidf_model_essay.get_feature_names(), list(te_tfidf_
model_essay.idf_)))
te_tfidf_model_essay = set(te_tfidf_model_essay.get_feature_names())
```

In [125]:
```
# TFIDF weighted Word2Vec for test essay
# compute average word2vec for each review.
te_tfidf_w2v_essay_vectors = []; # the avg-w2v for each sentence/review is sto
red in this list
for sentence in tqdm(X_test['essay'].values): # for each review/sentence
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/revie
w
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in te_tfidf_model_essay):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf v
alue((sentence.count(word)/len(sentence.split())))
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split
())) # getting the tfidf value for each word
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    te_tfidf_w2v_essay_vectors.append(vector)

print(len(te_tfidf_w2v_essay_vectors))
print(len(te_tfidf_w2v_essay_vectors[0]))
```

```
100%|████████████████████████████████████████████████████████████████████
███| 16500/16500 [01:03<00:00, 261.14it/s]

16500
300
```

2.3.8 Make Data Model Ready: project_title | TFIDF W2V

In [126]:
```
# S = ["abc def pqr", "def def def abc", "pqr pqr def"]
Tr_tfidf_model_title = TfidfVectorizer()
Tr_tfidf_model_title.fit(X_train['project_title'].values)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(Tr_tfidf_model_title.get_feature_names(), list(Tr_tfidf_
model_title.idf_)))
Tr_tfidf_model_title = set(Tr_tfidf_model_title.get_feature_names())
```

In [127]:
```
# TFIDF weighted Word2Vec for train title
# compute average word2vec for each review.
tr_tfidf_w2v_title_vectors = []; # the avg-w2v for each sentence/review is sto
red in this list
for sentence in tqdm(X_train['project_title'].values): # for each review/sente
nce
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/revie
w
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in Tr_tfidf_model_title):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf v
alue((sentence.count(word)/len(sentence.split())))
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split
())) # getting the tfidf value for each word
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    tr_tfidf_w2v_title_vectors.append(vector)

print(len(tr_tfidf_w2v_title_vectors))
print(len(tr_tfidf_w2v_title_vectors[0]))
```

```
100%|████████████████████████████████████████████████████████████████
██| 33500/33500 [00:00<00:00, 96237.65it/s]

33500
300
```

In [128]:
```
# S = ["abc def pqr", "def def def abc", "pqr pqr def"]
te_tfidf_model_title = TfidfVectorizer()
te_tfidf_model_title.fit(X_test['project_title'].values)
# we are converting a dictionary with word as a key, and the idf as a value
dictionary = dict(zip(te_tfidf_model_title.get_feature_names(), list(te_tfidf_
model_title.idf_)))
te_tfidf_model_title = set(te_tfidf_model_title.get_feature_names())
```

In [129]:
```
# TFIDF weighted Word2Vec for test title
# compute average word2vec for each review.
te_tfidf_w2v_title_vectors = []; # the avg-w2v for each sentence/review is sto
red in this list
for sentence in tqdm(X_test['project_title'].values): # for each review/senten
ce
    vector = np.zeros(300) # as word vectors are of zero length
    tf_idf_weight =0; # num of words with a valid vector in the sentence/revie
w
    for word in sentence.split(): # for each word in a review/sentence
        if (word in glove_words) and (word in te_tfidf_model_title):
            vec = model[word] # getting the vector for each word
            # here we are multiplying idf value(dictionary[word]) and the tf v
alue((sentence.count(word)/len(sentence.split())))
            tf_idf = dictionary[word]*(sentence.count(word)/len(sentence.split
())) # getting the tfidf value for each word
            vector += (vec * tf_idf) # calculating tfidf weighted w2v
            tf_idf_weight += tf_idf
    if tf_idf_weight != 0:
        vector /= tf_idf_weight
    te_tfidf_w2v_title_vectors.append(vector)

print(len(te_tfidf_w2v_title_vectors))
print(len(te_tfidf_w2v_title_vectors[0]))
```

```
100%|████████████████████████████████████████████████████████████████████
█| 16500/16500 [00:00<00:00, 96750.49it/s]

16500
300
```

2.4 Applying Random Forest

Apply Random Forest on different kind of featurization as mentioned in the instructions
For Every model that you work on make sure you do the step 2 and step 3 of instrucations

2.4.1 Applying Random Forests on BOW, SET 1

In [130]:
```
# Please write all the code with proper documentation
```

1. Set 1: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(BOW) + preprocessed_eassay (BOW)

In [131]:
```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
X_tr_bow = hstack((X_train_essay_bow, X_train_title_bow, X_train_school, X_tra
in_category, X_train_subcategory, X_train_grade, X_train_tea_pfx, X_train_prj_
res_sum, X_train_quantity_norm, X_train_TprevPrj_norm, X_train_price_norm)).to
csr()
X_te_bow = hstack((X_test_essay_bow, X_test_title_bow , X_test_school, X_test_
category, X_test_subcategory, X_test_grade, X_test_tea_pfx, X_test_prj_res_sum
, X_test_quantity_norm, X_test_TprevPrj_norm, X_test_price_norm)).tocsr()

print("Final Data matrix | BOW")
print(X_tr_bow.shape, y_train.shape)
print(X_te_bow.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix | BOW
(33500, 8420) (33500,)
(16500, 8420) (16500,)
====================================================================================
======================
```

In [133]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/m
achine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
from sklearn.ensemble import RandomForestClassifier

d_range=[1,5,10,50,100,500,1000]
n_est=[5,10,100,500]

param_grid=dict(max_depth=d_range,n_estimators=n_est)
print(param_grid)

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.Gr
idSearchCV.html
modelBow = GridSearchCV(RandomForestClassifier(class_weight="balanced"), param
_grid, scoring = 'f1', cv=5)
modelBow.fit(X_tr_bow, y_train)

print(modelBow.best_estimator_)
print(modelBow.score(X_te_bow, y_test))
```

```
{'max_depth': [1, 5, 10, 50, 100, 500, 1000], 'n_estimators': [5, 10, 100, 50
0]}
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=500, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.916026020106446
```

Please see below code, please correct it is the right approach or not, for drawing scatter graph.

Since we need to make graph on x,y and z. so taking the z value as (1,1),(2,2),(3,3),(4,4)

In [185]:
```
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html
#print(modelBow.cv_results_)
max_scoresBow=pd.DataFrame(modelBow.cv_results_).groupby(['param_n_estimators','param_max_depth']).max().unstack()[['mean_test_score','mean_train_score']]
#print(max_scoresBow.mean_train_score)
#print(max_scoresBow.mean_test_score)

print(type(max_scoresBow.mean_test_score))
#print("max_scoresBow",max_scoresBow)
#print(type(max_scoresBow))
#print(max_scoresBow.shape)

print("max_scoresBow.mean_test_score.shape")
print(max_scoresBow.mean_test_score)
print(max_scoresBow.mean_test_score.shape)
print(len(max_scoresBow.mean_test_score))
```

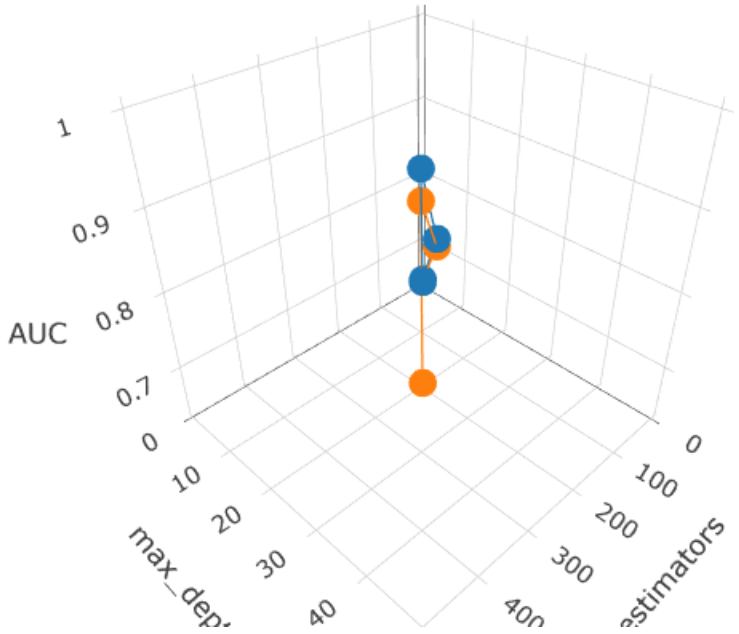
```
<class 'pandas.core.frame.DataFrame'>
max_scoresBow.mean_test_score.shape
param_max_depth         1         5         10        50        100   \
param_n_estimators
5                   0.639262  0.683435  0.733120  0.886347  0.899900
10                  0.733380  0.712060  0.764128  0.906308  0.911032
100                 0.707907  0.759216  0.821034  0.915993  0.916037
500                 0.726997  0.764103  0.823968  0.916060  0.916065

param_max_depth         500       1000
param_n_estimators
5                   0.899745  0.899308
10                  0.907963  0.908777
100                 0.916048  0.916025
500                 0.916083  0.916083
(4, 7)
4
```

In [276]:
```
#for i in range(max_scoresBow.mean_train_score.shape[0]):
#    if X_train_prj_res_sum['State_0'].isnull().iloc[i]:
#        nan_index_tr.append(i)
# https://stackoverflow.com/questions/16476924/how-to-iterate-over-rows-in-a-dataframe-in-pandas
#for i, row in max_scoresBow.mean_train_score.iterrows():
#    for j, column in row.iteritems():
#        #print(column)
#        List_tr.append(column)
#print(List_tr)
#len(List_tr)
#
#for i, row in max_scoresBow.mean_train_score.iterrows():
#    for j, column in row.iteritems():
#        print(#)
#        #print(column)
#        List_tr.append(column)
#print(List_tr)
len(List_tr)
List_tr=[]
dfTr=max_scoresBow.mean_train_score
print(len(dfTr))
for i in range(0, len(dfTr)):
    for j in range(0, len(dfTr)):
        #print(i,j)
        if i==j:
            #print(dfTr.iloc[i,j])
            List_tr.append(dfTr.iloc[i,j])
print(List_tr)
len(List_tr)

List_te=[]
dfTe=max_scoresBow.mean_test_score
print(len(dfTe))
for i in range(0, len(dfTe)):
    for j in range(0, len(dfTe)):
        #print(i,j)
        if i==j:
            #print(dfTe.iloc[i,j])
            List_te.append(dfTe.iloc[i,j])
print(List_te)
len(List_te)
```

```
4
[0.6433254212896172, 0.7237286455787841, 0.8627960694060791, 0.9999955878135409]
4
[0.6392618139052583, 0.7120600993631199, 0.8210338779118596, 0.9160596559355398]
```

Out[276]: 4

In [277]:
```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
import numpy as np

# https://plot.ly/python/3d-axes/
trace1 = go.Scatter3d(x=[5,10,100,500],y=[1,5,10,50,100,500,1000],z=List_tr, n
ame = 'train')
trace2 = go.Scatter3d(x=[5,10,100,500],y=[1,5,10,50,100,500,1000],z=List_te, n
ame = 'Cross validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```



Conclusion

For all the various values of n_estimators=500 and max_depth=500 is giving the best score for test data.

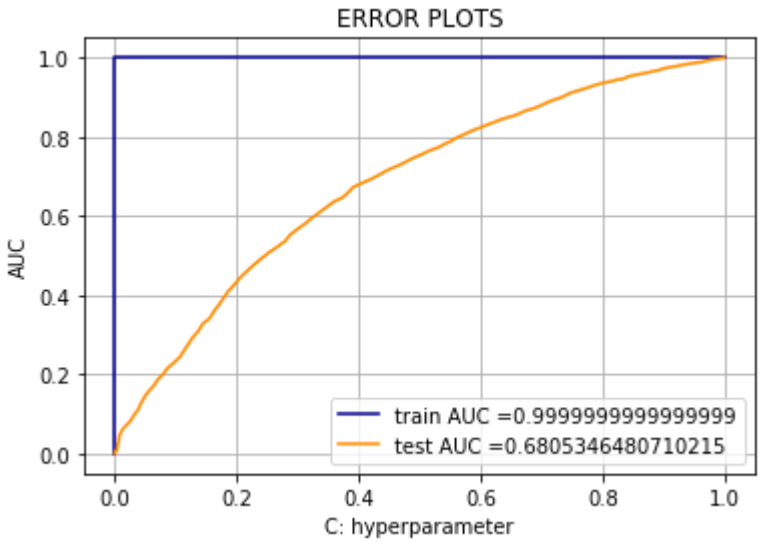
In [192]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
from sklearn.ensemble import RandomForestClassifier

d_range=[500]
n_est=[500]

param_grid=dict(max_depth=d_range,n_estimators=n_est)
print(param_grid)

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modelBowB = GridSearchCV(RandomForestClassifier(class_weight="balanced"), param_grid, scoring = 'f1', cv=5)
modelBowB.fit(X_tr_bow, y_train)

print(modelBowB.best_estimator_)
print(modelBowB.score(X_te_bow, y_test))
```

```
{'max_depth': [500], 'n_estimators': [500]}
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=500, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.9160616314596406
```

In [193]:
```
#best_tuned_parameters = [{'max_depth': [1], 'n_estimators' :[5]}]
```

In [194]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc

#modelBowB = GridSearchCV(RandomForestClassifier(), best_tuned_parameters)
#modelBowB.fit(X_tr_bow, y_train)

# roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of the positive class
# not the predicted outputs
#print(type(model.predict_proba(X_tr_bow)))
#print(model.predict_proba(X_tr_bow))
#print(model.predict_proba(X_tr_bow)[:,1])

y_train_bow_pred = modelBowB.predict_proba(X_tr_bow)[:,1]
y_test_bow_pred = modelBowB.predict_proba(X_te_bow)[:,1]

print(modelBowB.best_estimator_)
print(modelBowB.score(X_te_bow, y_test))

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_bow_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_bow_pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)),color='darkblue')
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)),color='darkorange')
plt.legend()
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("AUC PLOTS")
plt.grid(True)
plt.show()
```

```
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=500, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.9160616314596406
```



In [195]:
```
# we are writing our own function for predict, with defined thresould
# we will pick a threshold that will give the least fpr
def predict(proba, threshould, fpr, tpr):

    t = threshould[np.argmax(tpr*(1-fpr))]

    # (tpr*(1-fpr)) will be maximum if your fpr is very low and tpr is very hi
gh

    print("the maximum value of tpr*(1-fpr)", max(tpr*(1-fpr)), "for threshol
d", np.round(t,3))
    predictions = []
    for i in proba:
        if i>=t:
            predictions.append(1)
        else:
            predictions.append(0)
    return predictions
```

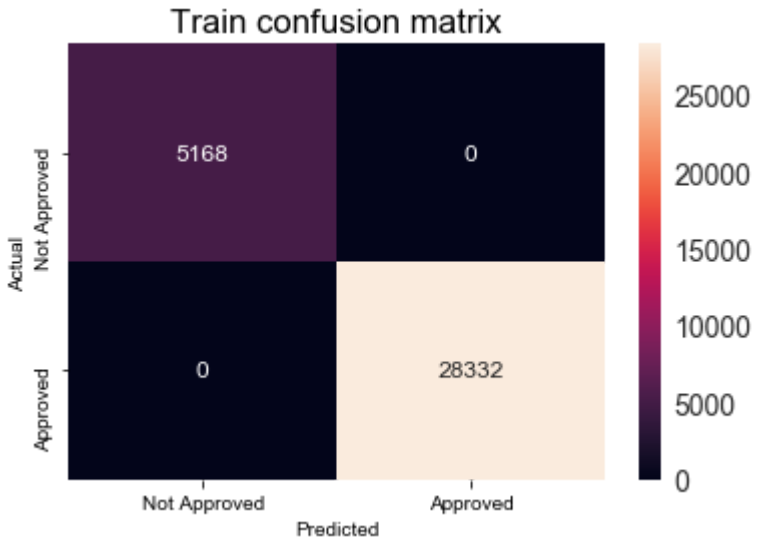
In [196]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
print("="*100)
from sklearn.metrics import confusion_matrix
print("Train confusion matrix")
print(confusion_matrix(y_train, predict(y_train_bow_pred, tr_thresholds, train_fpr, train_tpr)))
print("Test confusion matrix")
print(confusion_matrix(y_test, predict(y_test_bow_pred, te_thresholds, test_fpr, test_tpr)))
```

```
====================================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 1.0 for threshold 0.724
[[ 5168     0]
 [    0 28332]]
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.4096729676602091 for threshold 0.824
[[1549  997]
 [4558 9396]]
```

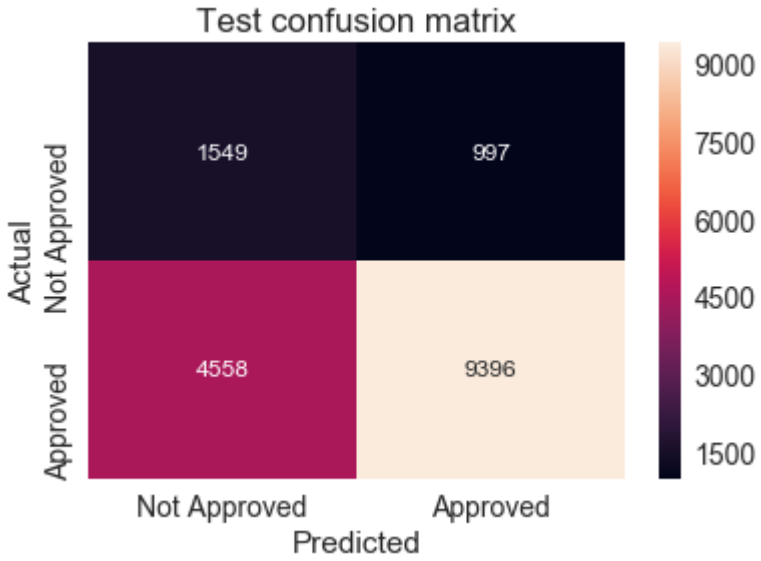
In [197]:
```
import seaborn as snTr
import seaborn as snTe
import pandas as pdH
import matplotlib.pyplot as pltTr
import matplotlib.pyplot as pltTe

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTr=confusion_matrix(y_train, predict(y_train_bow_pred, tr_thresholds, train_fpr, train_tpr))
df_cmTr = pdH.DataFrame(arrayTr,range(2),range(2))
#print(arrayTr)
# https://stackoverflow.com/questions/32723798/how-do-i-add-a-title-to-seaborn-heatmap
axTr = pltTr.axes()

snTr.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html

snTr.heatmap(df_cmTr, annot=True,annot_kws={"size": 12},fmt="d",ax=axTr)# font size, format in digit

labels=['Not Approved','Approved']
axTr.set_xticklabels(labels)
axTr.set_yticklabels(labels)
#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
pltTr.title("Train confusion matrix")
pltTr.xlabel("Predicted")
pltTr.ylabel("Actual")
pltTr.show()

# https://stackoverflow.com/questions/50947776/plot-two-seaborn-heatmap-graphs-side-by-side
#fig, ax =plt.subplots(1,1)

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTe=confusion_matrix(y_test, predict(y_test_bow_pred, te_thresholds, test_fpr, test_tpr))
df_cmTe = pdH.DataFrame(arrayTe,range(2),range(2))

axTe = pltTe.axes()

snTe.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html
snTe.heatmap(df_cmTe, annot=True,annot_kws={"size": 12},fmt="d",ax=axTe)# font size, format in digit

#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
axTe.set_xticklabels(labels)
axTe.set_yticklabels(labels)
pltTe.title("Test confusion matrix")
pltTe.xlabel("Predicted")
```

```
pltTe.ylabel("Actual")
pltTe.show()
```

```
the maximum value of tpr*(1-fpr) 1.0 for threshold 0.724
```




```
the maximum value of tpr*(1-fpr) 0.4096729676602091 for threshold 0.824
```



2.4.2 Applying Random Forests on TFIDF, SET 2

In [198]:
```
# Please write all the code with proper documentation
```

1. Set 2: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(TFIDF)+ preprocessed_eassay (TFIDF)

In [199]:
```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
X_tr_tfidf = hstack((X_train_text_tfidf, X_train_title_tfidf, X_train_school,
X_train_category, X_train_subcategory, X_train_grade, X_train_tea_pfx, X_train
_prj_res_sum, X_train_quantity_norm, X_train_TprevPrj_norm, X_train_price_norm
)).tocsr()
X_te_tfidf = hstack((X_test_text_tfidf, X_test_title_tfidf , X_test_school, X_
test_category, X_test_subcategory, X_test_grade, X_test_tea_pfx, X_test_prj_re
s_sum, X_test_quantity_norm, X_test_TprevPrj_norm, X_test_price_norm)).tocsr()

print("Final Data matrix | tfidf")
print(X_tr_tfidf.shape, y_train.shape)
print(X_te_tfidf.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix | tfidf
(33500, 8420) (33500,)
(16500, 8420) (16500,)
====================================================================================
======================
```

In [200]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/m
achine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
from sklearn.ensemble import RandomForestClassifier

d_range=[1,5,10,50,100,500,1000]
n_est=[5,10,100,500]

param_grid=dict(max_depth=d_range,n_estimators=n_est)
print(param_grid)

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.Gr
idSearchCV.html
modeltfidf = GridSearchCV(RandomForestClassifier(class_weight="balanced"), par
am_grid, scoring = 'f1', cv=5)
modeltfidf.fit(X_tr_tfidf, y_train)

print(modeltfidf.best_estimator_)
print(modeltfidf.score(X_te_tfidf, y_test))
```

```
{'max_depth': [1, 5, 10, 50, 100, 500, 1000], 'n_estimators': [5, 10, 100, 50
0]}
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=100, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=100, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.9160972404730618
```

Please see below code, please correct it is the right approach or not, for drawing scatter graph.

Since we need to make graph on x,y and z, so taking the z value as (1,1),(2,2),(3,3),(4,4)

In [201]:
```
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html
#print(modelBow.cv_results_)
max_scorestfidf=pd.DataFrame(modeltfidf.cv_results_).groupby(['param_n_estimators','param_max_depth']).max().unstack()[['mean_test_score','mean_train_score']]
#print(max_scoresBow.mean_train_score)
#print(max_scoresBow.mean_test_score)

print(type(max_scorestfidf.mean_test_score))
#print("max_scoresBow",max_scoresBow)
#print(type(max_scoresBow))
#print(max_scoresBow.shape)

print("max_scorestfidf.mean_test_score.shape")
print(max_scorestfidf.mean_test_score)
print(max_scorestfidf.mean_test_score.shape)
print(len(max_scorestfidf.mean_test_score))
```

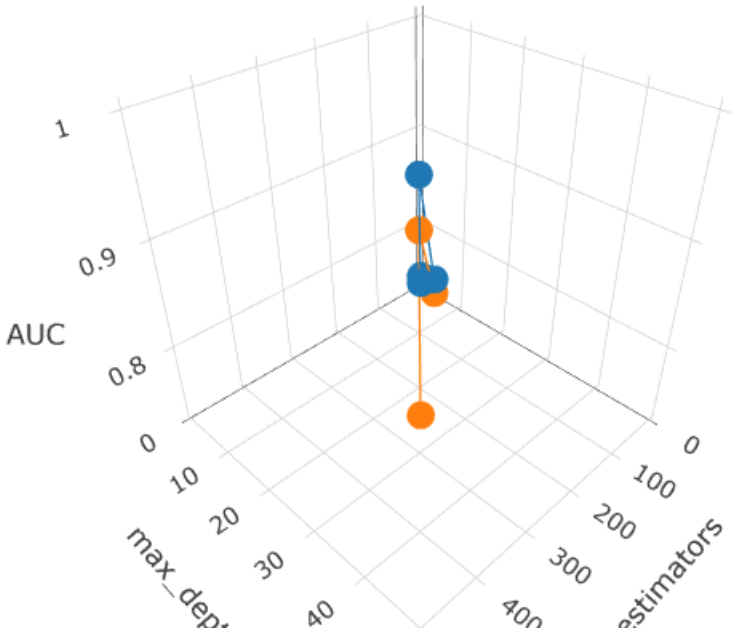
```
<class 'pandas.core.frame.DataFrame'>
max_scorestfidf.mean_test_score.shape
param_max_depth         1         5         10        50        100   \
param_n_estimators
5                   0.740151  0.706817  0.756500  0.890500  0.899878
10                  0.669269  0.732669  0.793690  0.908295  0.908818
100                 0.744911  0.773552  0.839256  0.916083  0.916185
500                 0.743360  0.785123  0.845177  0.916153  0.916103

param_max_depth         500       1000
param_n_estimators
5                   0.899723  0.899858
10                  0.907500  0.907176
100                 0.916121  0.916118
500                 0.916121  0.916121
(4, 7)
4
```

In [278]:
```
len(List_tr)
List_tr=[]
dfTr=max_scorestfidf.mean_train_score
print(len(dfTr))
for i in range(0, len(dfTr)):
    for j in range(0, len(dfTr)):
        #print(i,j)
        if i==j:
            #print(dfTr.iloc[i,j])
            List_tr.append(dfTr.iloc[i,j])
print(List_tr)
len(List_tr)

List_te=[]
dfTe=max_scorestfidf.mean_test_score
print(len(dfTe))
for i in range(0, len(dfTe)):
    for j in range(0, len(dfTe)):
        #print(i,j)
        if i==j:
            #print(dfTe.iloc[i,j])
            List_te.append(dfTe.iloc[i,j])
print(List_te)
len(List_te)
```

```
4
[0.7413448669630271, 0.7491163641375203, 0.8935810896963577, 1.0]
4
[0.7401506399473771, 0.7326686934390837, 0.8392557361217275, 0.91615298352474
77]
```

Out[278]: 4

In [279]:
```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
import numpy as np

# https://plot.ly/python/3d-axes/
trace1 = go.Scatter3d(x=[5,10,100,500],y=[1,5,10,50,100,500,1000],z=List_tr, n
ame = 'train')
trace2 = go.Scatter3d(x=[5,10,100,500],y=[1,5,10,50,100,500,1000],z=List_te, n
ame = 'Cross validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```



Conclusion

For all the various values of n_estimators=100 and max_depth=100 is giving the best score for test data.

In [204]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
from sklearn.ensemble import RandomForestClassifier

d_range=[100]
n_est=[100]

param_grid=dict(max_depth=d_range,n_estimators=n_est)
print(param_grid)

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modelTFIDFB = GridSearchCV(RandomForestClassifier(class_weight="balanced"), param_grid, scoring = 'f1', cv=5)
modelTFIDFB.fit(X_tr_tfidf, y_train)

print(modelTFIDFB.best_estimator_)
print(modelTFIDFB.score(X_te_tfidf, y_test))
```

```
{'max_depth': [100], 'n_estimators': [100]}
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=100, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=100, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.9161684514815058
```

In [205]:
```
#best_tuned_parameters = [{'max_depth': [1], 'n_estimators' :[5]}]
```

In [209]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc

#modelBowB = GridSearchCV(RandomForestClassifier(), best_tuned_parameters)
#modelBowB.fit(X_tr_bow, y_train)

# roc_auc_score(y_true, y_score) the 2nd parameter should be probability estimates of the positive class
# not the predicted outputs
#print(type(model.predict_proba(X_tr_bow)))
#print(model.predict_proba(X_tr_bow))
#print(model.predict_proba(X_tr_bow)[:,1])

y_train_tf_pred = modelTFIDFB.predict_proba(X_tr_tfidf)[:,1]
y_test_tf_pred = modelTFIDFB.predict_proba(X_te_tfidf)[:,1]

print(modelTFIDFB.best_estimator_)
print(modelTFIDFB.score(X_te_bow, y_test))

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_tf_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_tf_pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)),color='darkblue')
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)),color='darkorange')
plt.legend()
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("AUC Plot")
plt.grid(True)
plt.show()
```

```
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=100, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=100, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.9163985026597492
```

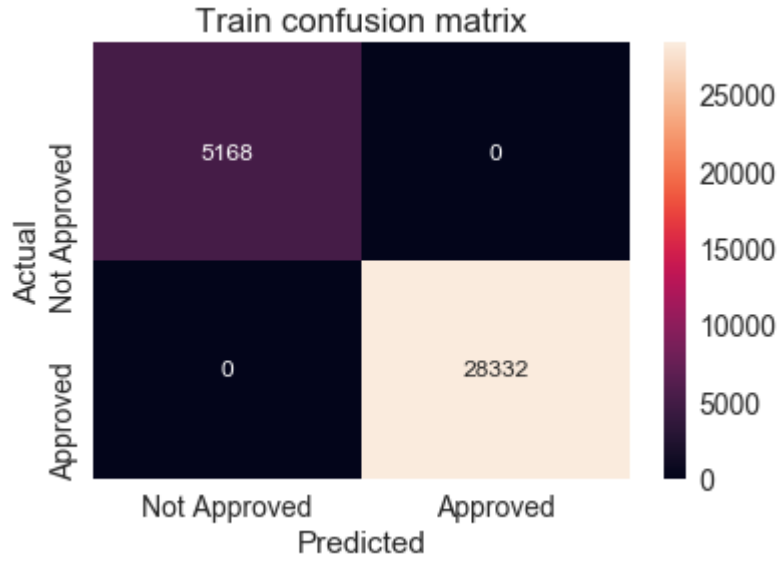
In [210]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_
matrix.html
print("="*100)
from sklearn.metrics import confusion_matrix
print("Train confusion matrix")
print(confusion_matrix(y_train, predict(y_train_tf_pred, tr_thresholds, train_
fpr, train_tpr)))
print("Test confusion matrix")
print(confusion_matrix(y_test, predict(y_test_tf_pred, te_thresholds, test_fpr
, test_tpr)))
```

```
====================================================================================
======================
Train confusion matrix
the maximum value of tpr*(1-fpr) 1.0 for threshold 0.635
[[ 5168     0]
 [    0 28332]]
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.3677595254343161 for threshold 0.821
[[1615  931]
 [5864 8090]]
```

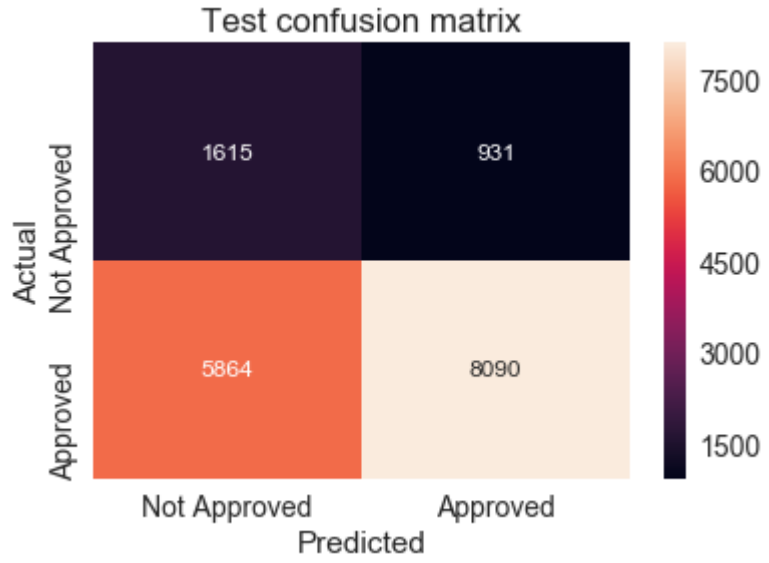
In [211]:
```
import seaborn as snTr
import seaborn as snTe
import pandas as pdH
import matplotlib.pyplot as pltTr
import matplotlib.pyplot as pltTe

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTr=confusion_matrix(y_train, predict(y_train_tf_pred, tr_thresholds, train_fpr, train_tpr))
df_cmTr = pdH.DataFrame(arrayTr,range(2),range(2))
#print(arrayTr)
# https://stackoverflow.com/questions/32723798/how-do-i-add-a-title-to-seaborn-heatmap
axTr = pltTr.axes()

snTr.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html

snTr.heatmap(df_cmTr, annot=True,annot_kws={"size": 12},fmt="d",ax=axTr)# font size, format in digit

labels=['Not Approved','Approved']
axTr.set_xticklabels(labels)
axTr.set_yticklabels(labels)
#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
pltTr.title("Train confusion matrix")
pltTr.xlabel("Predicted")
pltTr.ylabel("Actual")
pltTr.show()

# https://stackoverflow.com/questions/50947776/plot-two-seaborn-heatmap-graphs-side-by-side
#fig, ax =plt.subplots(1,1)

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTe=confusion_matrix(y_test, predict(y_test_tf_pred, te_thresholds, test_fpr, test_tpr))
df_cmTe = pdH.DataFrame(arrayTe,range(2),range(2))

axTe = pltTe.axes()

snTe.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html
snTe.heatmap(df_cmTe, annot=True,annot_kws={"size": 12},fmt="d",ax=axTe)# font size, format in digit

#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
axTe.set_xticklabels(labels)
axTe.set_yticklabels(labels)
pltTe.title("Test confusion matrix")
pltTe.xlabel("Predicted")
```

```
pltTe.ylabel("Actual")
plt.show()
```

```
the maximum value of tpr*(1-fpr) 1.0 for threshold 0.635
```




```
the maximum value of tpr*(1-fpr) 0.3677595254343161 for threshold 0.821
```



2.4.3 Applying Random Forests on AVG W2V, SET 3

In [0]:
```
# Please write all the code with proper documentation
```

1. Set 3: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(AVG W2V)+ preprocessed_eassay (AVG W2V)

In [212]:
```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
X_tr_avgW2V = hstack((X_train_essay_avg_w2v, X_train_title_avg_w2v, X_train_sc
hool, X_train_category, X_train_subcategory, X_train_grade, X_train_tea_pfx, X
_train_prj_res_sum, X_train_quantity_norm, X_train_TprevPrj_norm, X_train_pric
e_norm)).tocsr()
X_te_avgW2V = hstack((X_test_essay_avg_w2v, X_test_title_avg_w2v , X_test_scho
ol, X_test_category, X_test_subcategory, X_test_grade, X_test_tea_pfx, X_test_
prj_res_sum, X_test_quantity_norm, X_test_TprevPrj_norm, X_test_price_norm)).t
ocsr()

print("Final Data matrix | Avg W2V")
print(X_tr_avgW2V.shape, y_train.shape)
print(X_te_avgW2V.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix | Avg W2V
(33500, 615) (33500,)
(16500, 615) (16500,)
====================================================================================
======================
```

In [215]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
#from sklearn.grid_search import GridSearchCV
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
from sklearn.tree import DecisionTreeClassifier

d_range=[1,5,10,50,100,500,1000]
n_est=[5,10,100,500]

param_grid=dict(max_depth=d_range,n_estimators=n_est)
print(param_grid)

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modelavgW2V = GridSearchCV(RandomForestClassifier(class_weight="balanced"), param_grid, scoring='f1', cv=5)
modelavgW2V.fit(X_tr_avgW2V, y_train)

print(modelavgW2V.best_estimator_)
print(modelavgW2V.score(X_te_avgW2V, y_test))
```

```
{'max_depth': [1, 5, 10, 50, 100, 500, 1000], 'n_estimators': [5, 10, 100, 500]}
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=50, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.916393011953238
```

Please see below code, please correct it is the right approach or not, for drawing scatter graph.

Since we need to make graph on x,y and z. so taking the z value as (1,1),(2,2),(3,3),(4,4)

In [218]:
```
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html
max_scoreAvgW2v=pd.DataFrame(modelavgW2V.cv_results_).groupby(['param_n_estimators','param_max_depth']).max().unstack()[['mean_test_score','mean_train_score']]

print(type(max_scoreAvgW2v.mean_test_score))

print("max_scoreAvgW2v.mean_test_score.shape")
print(max_scoreAvgW2v.mean_test_score)
print(max_scoreAvgW2v.mean_test_score.shape)
print(len(max_scoreAvgW2v.mean_test_score))
```

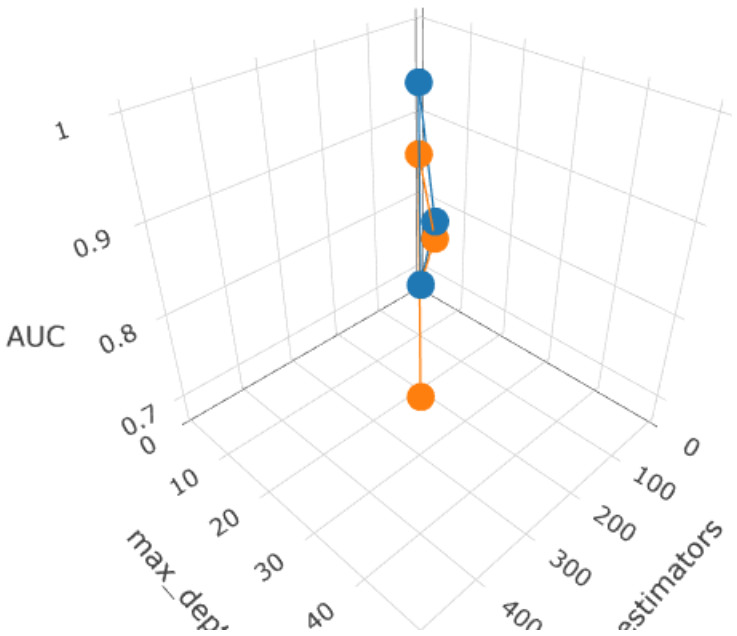
```
<class 'pandas.core.frame.DataFrame'>
max_scoreAvgW2v.mean_test_score.shape
param_max_depth         1         5         10        50        100   \
param_n_estimators
5                   0.678878  0.742995  0.829288  0.904271  0.904372
10                  0.726506  0.758323  0.856447  0.911291  0.910622
100                 0.721345  0.784804  0.896980  0.916314  0.916331
500                 0.721371  0.785486  0.899627  0.916366  0.916349

param_max_depth         500       1000
param_n_estimators
5                   0.903648  0.904581
10                  0.910849  0.910543
100                 0.916326  0.916278
500                 0.916349  0.916349
(4, 7)
4
```

In [280]:
```
len(List_tr)
List_tr=[]
dfTr=max_scoreAvgW2v.mean_train_score
print(len(dfTr))
for i in range(0, len(dfTr)):
    for j in range(0, len(dfTr)):
        #print(i,j)
        if i==j:
            #print(dfTr.iloc[i,j])
            List_tr.append(dfTr.iloc[i,j])
print(List_tr)
len(List_tr)

List_te=[]
dfTe=max_scoreAvgW2v.mean_test_score
print(len(dfTe))
for i in range(0, len(dfTe)):
    for j in range(0, len(dfTe)):
        #print(i,j)
        if i==j:
            #print(dfTe.iloc[i,j])
            List_te.append(dfTe.iloc[i,j])
print(List_te)
len(List_te)
```

```
4
[0.6796267921082513, 0.7811940707255942, 0.97091467524057, 0.999973527270563
9]
4
[0.6788782064253397, 0.7583227248448233, 0.8969802472089955, 0.91636610001822
03]
```

Out[280]: 4

In [281]:
```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
import numpy as np

# https://plot.ly/python/3d-axes/
trace1 = go.Scatter3d(x=[5,10,100,500],y=[1,5,10,50,100,500,1000],z=List_tr, n
ame = 'train')
trace2 = go.Scatter3d(x=[5,10,100,500],y=[1,5,10,50,100,500,1000],z=List_te, n
ame = 'Cross validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```



Conclusion

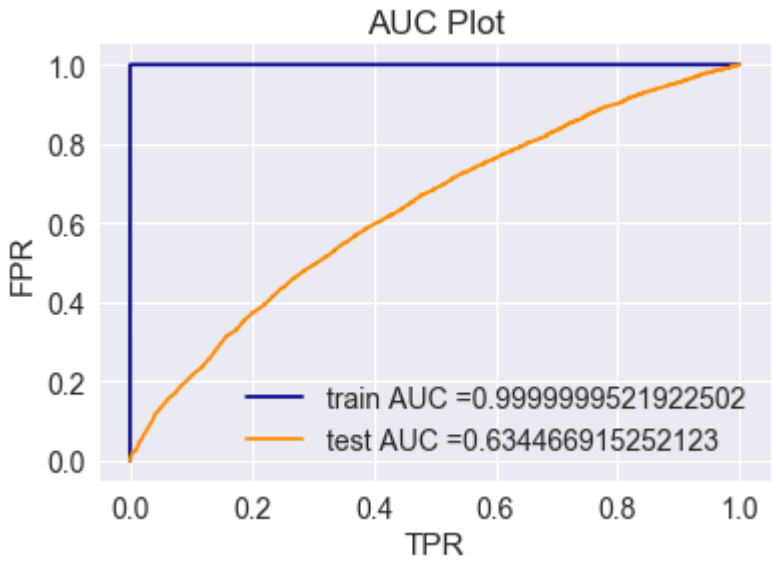
For all the various values of n_estimators=500 and max_depth=50 is giving the best score for test data.

In [221]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
from sklearn.ensemble import RandomForestClassifier

d_range=[50]
n_est=[500]

param_grid=dict(max_depth=d_range,n_estimators=n_est)
print(param_grid)

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modelavgW2VB = GridSearchCV(RandomForestClassifier(class_weight="balanced"), param_grid, scoring = 'roc_auc', cv=5)
modelavgW2VB.fit(X_tr_avgW2V, y_train)

print(modelavgW2VB.best_estimator_)
print(modelavgW2VB.score(X_te_avgW2V, y_test))
```

```
{'max_depth': [50], 'n_estimators': [500]}
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=50, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.634466915252123
```

In [222]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc

y_train_avgW2V_pred = modelavgW2VB.predict_proba(X_tr_avgW2V)[:,1]
y_test_avgW2V_pred = modelavgW2VB.predict_proba(X_te_avgW2V)[:,1]

print(modelavgW2VB.best_estimator_)
print(modelavgW2VB.score(X_te_avgW2V, y_test))

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_avgW2V_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_avgW2V_pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)),color='darkblue')
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)),color='darkorange')
plt.legend()
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("AUC Plot")
plt.grid(True)
plt.show()
```

```
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=50, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.634466915252123
```



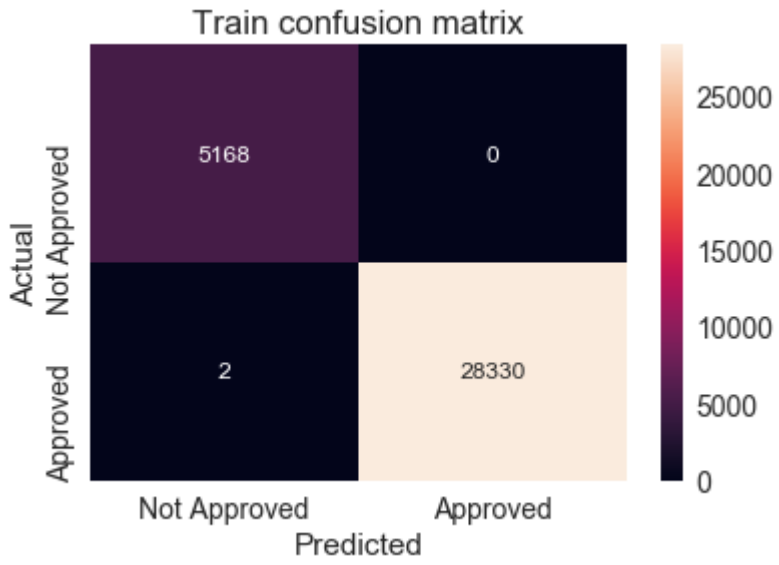
In [223]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
print("="*100)
from sklearn.metrics import confusion_matrix
print("Train confusion matrix")
print(confusion_matrix(y_train, predict(y_train_avgW2V_pred, tr_thresholds, train_fpr, train_tpr)))
print("Test confusion matrix")
print(confusion_matrix(y_test, predict(y_test_avgW2V_pred, te_thresholds, test_fpr, test_tpr)))
```

```
====================================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 0.9999294084427502 for threshold 0.76
[[ 5168     0]
 [    2 28330]]
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.35928557652283827 for threshold 0.85
[[1561  985]
 [5777 8177]]
```

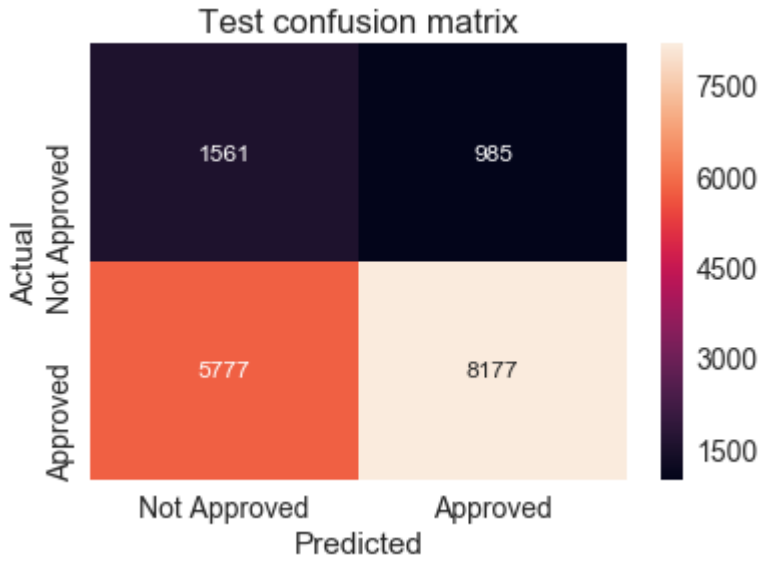
In [224]:
```
import seaborn as snTr
import seaborn as snTe
import pandas as pdH
import matplotlib.pyplot as pltTr
import matplotlib.pyplot as pltTe

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTr=confusion_matrix(y_train, predict(y_train_avgW2V_pred, tr_thresholds, train_fpr, train_tpr))
df_cmTr = pdH.DataFrame(arrayTr,range(2),range(2))
#print(arrayTr)
# https://stackoverflow.com/questions/32723798/how-do-i-add-a-title-to-seaborn-heatmap
axTr = pltTr.axes()

snTr.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html

snTr.heatmap(df_cmTr, annot=True,annot_kws={"size": 12},fmt="d",ax=axTr)# font size, format in digit

labels=['Not Approved','Approved']
axTr.set_xticklabels(labels)
axTr.set_yticklabels(labels)
#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
pltTr.title("Train confusion matrix")
pltTr.xlabel("Predicted")
pltTr.ylabel("Actual")
pltTr.show()

# https://stackoverflow.com/questions/50947776/plot-two-seaborn-heatmap-graphs-side-by-side
```

```
#fig, ax =plt.subplots(1,1)

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTe=confusion_matrix(y_test, predict(y_test_avgW2V_pred, te_thresholds, test_fpr, test_tpr))
df_cmTe = pdH.DataFrame(arrayTe,range(2),range(2))

axTe = pltTe.axes()

snTe.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html
snTe.heatmap(df_cmTe, annot=True,annot_kws={"size": 12},fmt="d",ax=axTe)# font size, format in digit

#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
axTe.set_xticklabels(labels)
axTe.set_yticklabels(labels)
pltTe.title("Test confusion matrix")
pltTe.xlabel("Predicted")
pltTe.ylabel("Actual")
pltTe.show()
```

```
the maximum value of tpr*(1-fpr) 0.9999294084427502 for threshold 0.76
```




```
the maximum value of tpr*(1-fpr) 0.35928557652283827 for threshold 0.85
```



2.4.4 Applying Random Forests on TFIDF W2V, SET 4

```
In [0]: # Please write all the code with proper documentation
```

1. Set 4: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(TFIDF W2V)+ preprocessed_eassay (TFIDF W2V)

In [216]:
```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
X_tr_tfidf_W2V = hstack((tr_tfidf_w2v_essay_vectors, tr_tfidf_w2v_title_vectors, X_train_school, X_train_category, X_train_subcategory, X_train_grade, X_train_tea_pfx, X_train_prj_res_sum, X_train_quantity_norm, X_train_TprevPrj_norm, X_train_price_norm)).tocsr()
X_te_tfidf_W2V = hstack((te_tfidf_w2v_essay_vectors, te_tfidf_w2v_title_vectors , X_test_school, X_test_category, X_test_subcategory, X_test_grade, X_test_tea_pfx, X_test_prj_res_sum, X_test_quantity_norm, X_test_TprevPrj_norm, X_test_price_norm)).tocsr()

print("Final Data matrix | TFIDF W2V")
print(X_tr_tfidf_W2V.shape, y_train.shape)
print(X_te_tfidf_W2V.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix | TFIDF W2V
(33500, 615) (33500,)
(16500, 615) (16500,)
====================================================================================================
```

In [217]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
from sklearn.ensemble import RandomForestClassifier

d_range=[1,5,10,50,100,500,1000]
n_est=[5,10,100,500]

param_grid=dict(max_depth=d_range,n_estimators=n_est)
print(param_grid)

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modeltfidf_W2V = GridSearchCV(RandomForestClassifier(class_weight="balanced"), param_grid, scoring = 'f1', cv=5)
modeltfidf_W2V.fit(X_tr_tfidf_W2V, y_train)

print(modeltfidf_W2V.best_estimator_)
print(modeltfidf_W2V.score(X_te_tfidf_W2V, y_test))
```

```
{'max_depth': [1, 5, 10, 50, 100, 500, 1000], 'n_estimators': [5, 10, 100, 500]}
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=50, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.9163574266854947
```

Please see below code, please correct it is the right approach or not, for drawing scatter graph.

Since we need to make graph on x,y and z. so taking the z value as (1,1),(2,2),(3,3),(4,4)

In [225]:
```
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html
max_scoreTfidfW2v=pd.DataFrame(modeltfidf_W2V.cv_results_).groupby(['param_n_estimators','param_max_depth']).max().unstack()[['mean_test_score','mean_train_score']]

print(type(max_scoreTfidfW2v.mean_test_score))

print("max_scoreTfidfW2v.mean_test_score.shape")
print(max_scoreTfidfW2v.mean_test_score)
print(max_scoreTfidfW2v.mean_test_score.shape)
print(len(max_scoreTfidfW2v.mean_test_score))
```

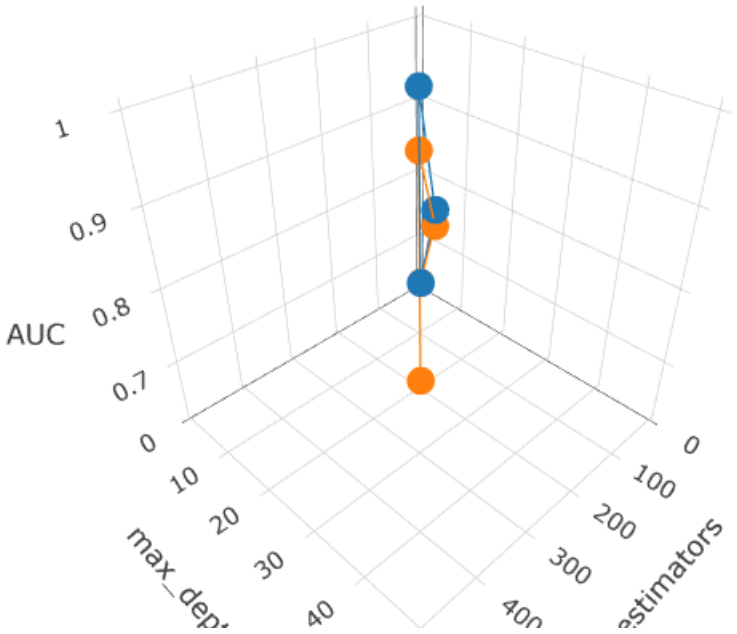
```
<class 'pandas.core.frame.DataFrame'>
max_scoreTfidfW2v.mean_test_score.shape
param_max_depth         1         5         10        50        100   \
param_n_estimators
5                  0.628430  0.713898  0.809269  0.903589  0.903952
10                 0.707324  0.736601  0.844648  0.909958  0.909866
100                0.708443  0.765244  0.882778  0.916328  0.916296
500                0.707688  0.768370  0.889217  0.916349  0.916331

param_max_depth         500       1000
param_n_estimators
5                  0.902961  0.904597
10                 0.909933  0.910308
100                0.916331  0.916296
500                0.916296  0.916331
(4, 7)
4
```

In [282]:
```
len(List_tr)
List_tr=[]
dfTr=max_scoreTfidfW2v.mean_train_score
print(len(dfTr))
for i in range(0, len(dfTr)):
    for j in range(0, len(dfTr)):
        #print(i,j)
        if i==j:
            #print(dfTr.iloc[i,j])
            List_tr.append(dfTr.iloc[i,j])
print(List_tr)
len(List_tr)

List_te=[]
dfTe=max_scoreTfidfW2v.mean_test_score
print(len(dfTe))
for i in range(0, len(dfTe)):
    for j in range(0, len(dfTe)):
        #print(i,j)
        if i==j:
            #print(dfTe.iloc[i,j])
            List_te.append(dfTe.iloc[i,j])
print(List_te)
len(List_te)
```

```
4
[0.6278711350694802, 0.7610451527112477, 0.9596706589579833, 0.99998676363528
84]
4
[0.6284299283596972, 0.7366007288833537, 0.8827776645700651, 0.91634856719199
38]
```

Out[282]: 4

In [283]:
```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
import numpy as np

# https://plot.ly/python/3d-axes/
trace1 = go.Scatter3d(x=[5,10,100,500],y=[1,5,10,50,100,500,1000],z=List_tr, n
ame = 'train')
trace2 = go.Scatter3d(x=[5,10,100,500],y=[1,5,10,50,100,500,1000],z=List_te, n
ame = 'Cross validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```



Conclusion

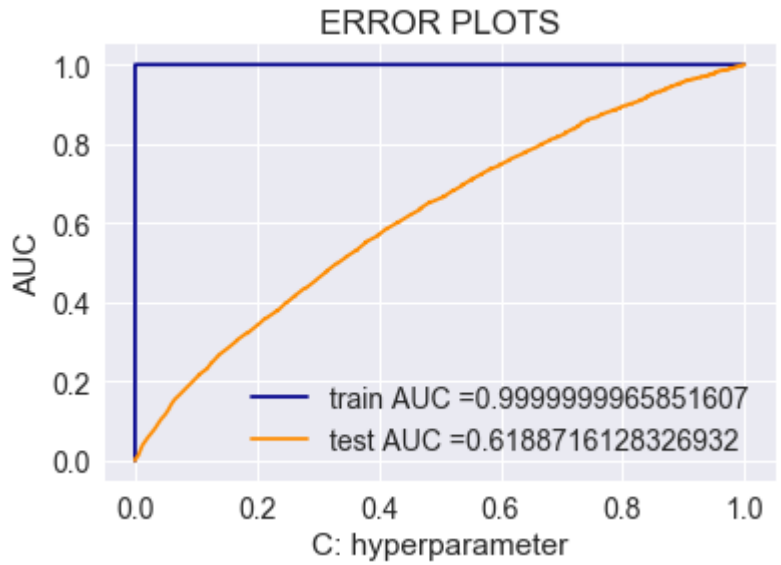
For all the various values of min_samples_split=500 and max_depth=50 is giving the best score for test data.

In [228]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
from sklearn.ensemble import RandomForestClassifier

d_range=[50]
n_est=[500]

param_grid=dict(max_depth=d_range,n_estimators=n_est)
print(param_grid)

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modeltfidf_W2VB = GridSearchCV(RandomForestClassifier(class_weight="balanced"
), param_grid, scoring = 'f1', cv=5)
modeltfidf_W2VB.fit(X_tr_tfidf_W2V, y_train)

print(modeltfidf_W2VB.best_estimator_)
print(modeltfidf_W2VB.score(X_te_tfidf_W2V, y_test))
```

```
{'max_depth': [50], 'n_estimators': [500]}
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=50, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.9162862491379028
```

In [229]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc

y_train_tfidf_w2v_pred = modeltfidf_W2VB.predict_proba(X_tr_tfidf_W2V)[:,1]
y_test_tfidf_w2v_pred = modeltfidf_W2VB.predict_proba(X_te_tfidf_W2V)[:,1]

print(modeltfidf_W2VB.best_estimator_)
print(modeltfidf_W2VB.score(X_te_tfidf_W2V, y_test))

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_tfidf_w2v_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_tfidf_w2v_pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)),color='darkblue')
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)),color='darkorange')
plt.legend()
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("AUC PLOTS")
plt.grid(True)
plt.show()
```

```
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='gini', max_depth=50, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=500, n_jobs=1, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
0.9162862491379028
```



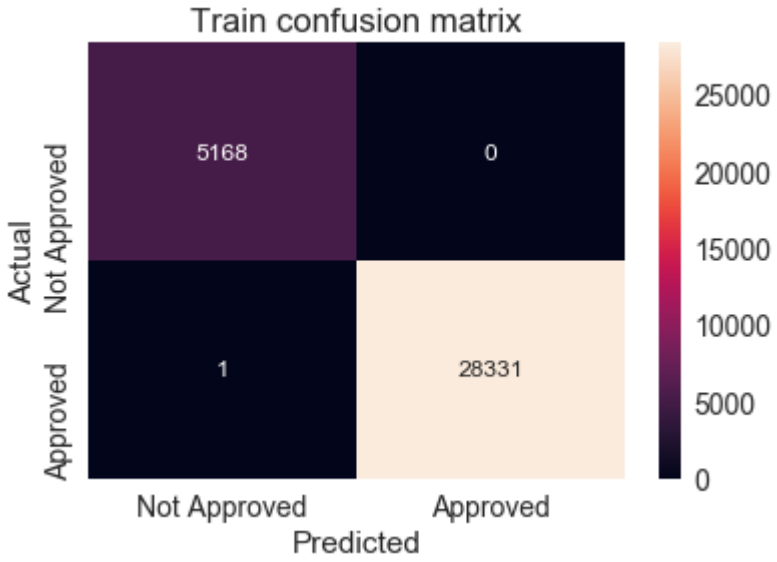
In [230]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
print("="*100)
from sklearn.metrics import confusion_matrix
print("Train confusion matrix")
print(confusion_matrix(y_train, predict(y_train_tfidf_w2v_pred, tr_thresholds, train_fpr, train_tpr)))
print("Test confusion matrix")
print(confusion_matrix(y_test, predict(y_test_tfidf_w2v_pred, te_thresholds, test_fpr, test_tpr)))
```

```
====================================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 0.9999647042213752 for threshold 0.758
[[ 5168     0]
 [    1 28331]]
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.3443707587752419 for threshold 0.848
[[1503 1043]
 [5814 8140]]
```

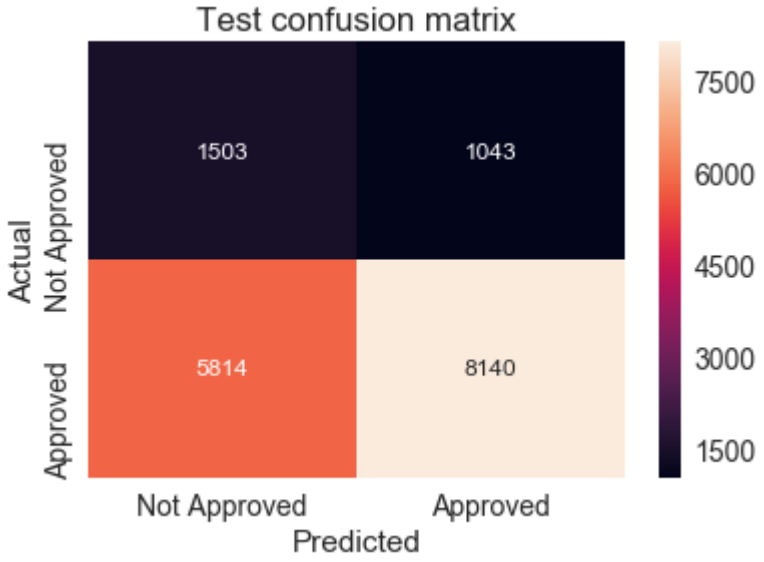
In [231]:
```
import seaborn as snTr
import seaborn as snTe
import pandas as pdH
import matplotlib.pyplot as pltTr
import matplotlib.pyplot as pltTe

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTr=confusion_matrix(y_train, predict(y_train_tfidf_w2v_pred, tr_thresholds, train_fpr, train_tpr))
df_cmTr = pdH.DataFrame(arrayTr,range(2),range(2))
#print(arrayTr)
# https://stackoverflow.com/questions/32723798/how-do-i-add-a-title-to-seaborn-heatmap
axTr = pltTr.axes()

snTr.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html

snTr.heatmap(df_cmTr, annot=True,annot_kws={"size": 12},fmt="d",ax=axTr)# font size, format in digit

labels=['Not Approved','Approved']
axTr.set_xticklabels(labels)
axTr.set_yticklabels(labels)
#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
pltTr.title("Train confusion matrix")
pltTr.xlabel("Predicted")
pltTr.ylabel("Actual")
pltTr.show()

# https://stackoverflow.com/questions/50947776/plot-two-seaborn-heatmap-graphs-side-by-side
#fig, ax =plt.subplots(1,1)

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTe=confusion_matrix(y_test, predict(y_test_tfidf_w2v_pred, te_thresholds, test_fpr, test_tpr))
df_cmTe = pdH.DataFrame(arrayTe,range(2),range(2))

axTe = pltTe.axes()

snTe.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html
snTe.heatmap(df_cmTe, annot=True,annot_kws={"size": 12},fmt="d",ax=axTe)# font size, format in digit

#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
axTe.set_xticklabels(labels)
axTe.set_yticklabels(labels)
pltTe.title("Test confusion matrix")
pltTe.xlabel("Predicted")
```

```
pltTe.ylabel("Actual")
pltTe.show()
```

```
the maximum value of tpr*(1-fpr) 0.9999647042213752 for threshold 0.758
```




```
the maximum value of tpr*(1-fpr) 0.3443707587752419 for threshold 0.848
```



2.5 Applying GBDT

Apply GBDT on different kind of featurization as mentioned in the instructions
For Every model that you work on make sure you do the step 2 and step 3 of instrucations

2.5.1 Applying XGBOOST on BOW, SET 1

In [0]:
```
# Please write all the code with proper documentation
```

1. Set 1: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(BOW) + preprocessed_eassay (BOW)

In [236]:
```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
X_tr_bow_XG = hstack((X_train_essay_bow, X_train_title_bow, X_train_school, X_
train_category, X_train_subcategory, X_train_grade, X_train_tea_pfx, X_train_p
rj_res_sum, X_train_quantity_norm, X_train_TprevPrj_norm, X_train_price_norm))
.tocsr()
X_te_bow_XG = hstack((X_test_essay_bow, X_test_title_bow , X_test_school, X_te
st_category, X_test_subcategory, X_test_grade, X_test_tea_pfx, X_test_prj_res_
sum, X_test_quantity_norm, X_test_TprevPrj_norm, X_test_price_norm)).tocsr()

print("Final Data matrix | BOW")
print(X_tr_bow_XG.shape, y_train.shape)
print(X_te_bow_XG.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix | BOW
(33500, 8420) (33500,)
(16500, 8420) (16500,)
====================================================================================
======================
```

In [237]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
import xgboost as xgb


xgb_model = xgb.XGBClassifier()

param_grid = {'max_depth': [5,7,15,25],
              'silent': [1],
              'n_estimators': [5,7,15,25], #number of trees, change it to 1000 for better results
             }

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modelBowXB = GridSearchCV(xgb_model, param_grid, scoring = 'roc_auc', cv=5)
modelBowXB.fit(X_tr_bow_XG, y_train)

print(modelBowXB.best_estimator_)
print(modelBowXB.score(X_te_bow_XG, y_test))
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.646336419484467
```

Please see below code, please correct it is the right approach or not, for drawing scatter graph.

Since we need to make graph on x,y and z. so taking the z value as (1,1),(2,2),(3,3),(4,4)

In [239]:
```
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html
#print(modelBow.cv_results_)
max_scoresBowXB=pd.DataFrame(modelBowXB.cv_results_).groupby(['param_n_estimators','param_max_depth']).max().unstack()[['mean_test_score','mean_train_score']]
#print(max_scoresBow.mean_train_score)
#print(max_scoresBow.mean_test_score)

print(type(max_scoresBowXB.mean_test_score))
#print("max_scoresBow",max_scoresBow)
#print(type(max_scoresBow))
#print(max_scoresBow.shape)

print("max_scoresBowXB.mean_test_score.shape")
print(max_scoresBowXB.mean_test_score)
print(max_scoresBowXB.mean_test_score.shape)
print(len(max_scoresBowXB.mean_test_score))
```

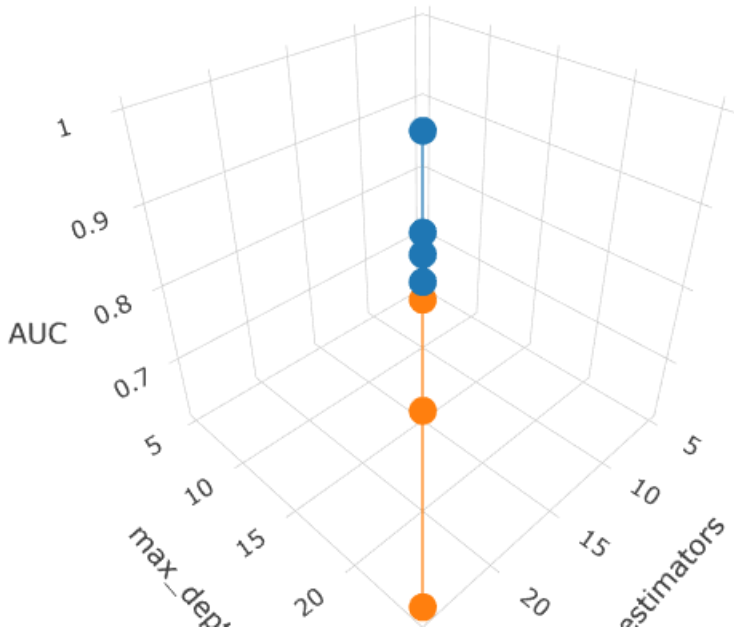
```
<class 'pandas.core.frame.DataFrame'>
max_scoresBowXB.mean_test_score.shape
param_max_depth           5         7         15        25
param_n_estimators
5                  0.623810  0.624415  0.600538  0.584509
7                  0.627636  0.631711  0.608793  0.592128
15                 0.642808  0.648868  0.628020  0.614610
25                 0.658765  0.666872  0.656191  0.644252
(4, 4)
4
```

In [284]:
```
List_tr_BowXB=[]
dfTr=max_scoresBowXB.mean_train_score
print(len(dfTr))
for i in range(0, len(dfTr)):
    for j in range(0, len(dfTr)):
        if i==j:
            List_tr_BowXB.append(dfTr.iloc[i,j])
print(List_tr_BowXB)
len(List_tr_BowXB)

List_te_BowXB=[]
dfTe=max_scoresBowXB.mean_test_score
print(len(dfTe))
for i in range(0, len(dfTe)):
    for j in range(0, len(dfTe)):
        if i==j:
            List_te_BowXB.append(dfTe.iloc[i,j])
print(List_te_BowXB)
len(List_te_BowXB)
```

```
4
[0.6709236345004064, 0.7398329655230913, 0.9821025625641783, 0.99999555327365
29]
4
[0.6238098125564003, 0.6317105264760206, 0.6280202266338053, 0.64425211335429
02]
```

Out[284]: 4

In [285]:
```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
import numpy as np

# https://plot.ly/python/3d-axes/
trace1 = go.Scatter3d(x=[5,7,15,25],y=[5,7,15,25],z=List_tr_BowXB, name = 'tra
in')
trace2 = go.Scatter3d(x=[5,7,15,25],y=[5,7,15,25],z=List_te_BowXB, name = 'Cro
ss validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```



Conclusion

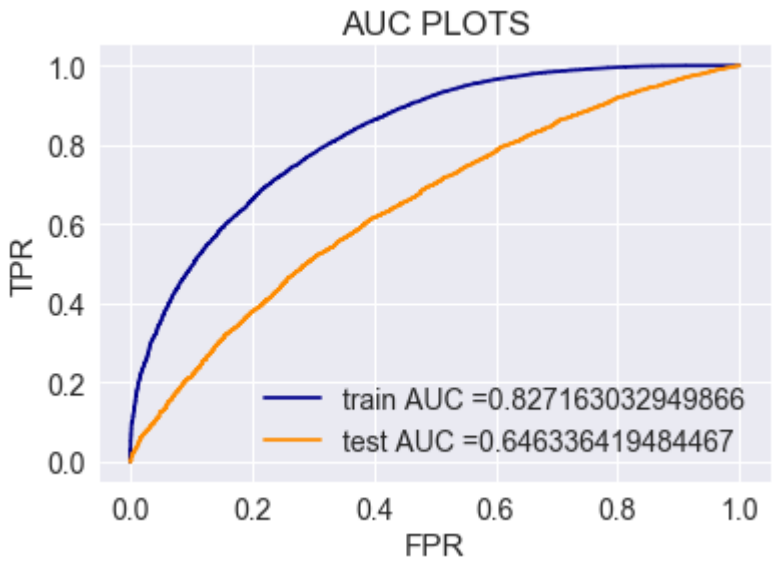
For all the various values of n_estimators=25 and max_depth=7 is giving the best score for test data.

In [242]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
import xgboost as xgb


xgb_model = xgb.XGBClassifier()

param_grid = {'max_depth': [7],
              'silent': [1],
              'n_estimators': [25], #number of trees, change it to 1000 for better results
             }

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modelBowXBb = GridSearchCV(xgb_model, param_grid, scoring = 'roc_auc', cv=5)
modelBowXBb.fit(X_tr_bow_XG, y_train)

print(modelBowXBb.best_estimator_)
print(modelBowXBb.score(X_te_bow_XG, y_test))
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.646336419484467
```

In [243]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc

y_train_bow_XB_pred = modelBowXBb.predict_proba(X_tr_bow_XG)[:,1]
y_test_bow_XB_pred = modelBowXBb.predict_proba(X_te_bow_XG)[:,1]

print(modelBowXBb.best_estimator_)
print(modelBowXBb.score(X_te_bow_XG, y_test))

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_bow_XB_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_bow_XB_pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)),color='darkblue')
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)),color='darkorange')
plt.legend()
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("AUC PLOTS")
plt.grid(True)
plt.show()
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.646336419484467
```



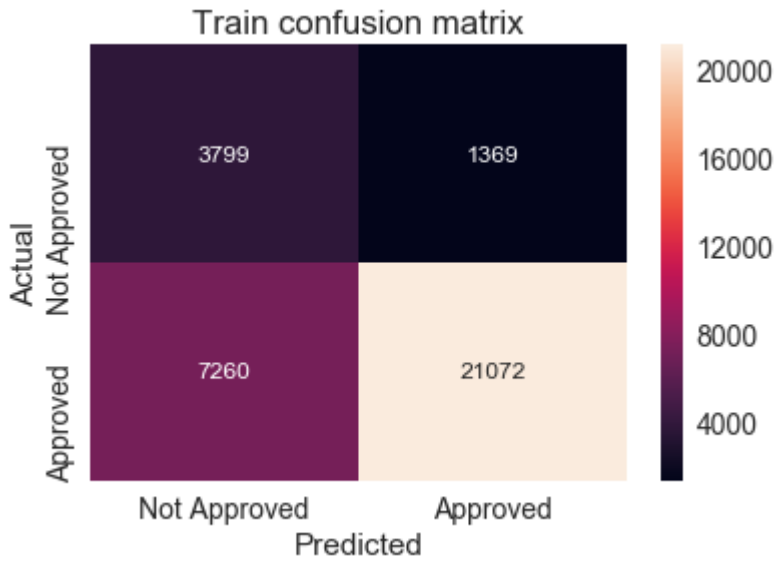
In [244]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
print("="*100)
from sklearn.metrics import confusion_matrix
print("Train confusion matrix")
print(confusion_matrix(y_train, predict(y_train_bow_XB_pred, tr_thresholds, train_fpr, train_tpr)))
print("Test confusion matrix")
print(confusion_matrix(y_test, predict(y_test_bow_XB_pred, te_thresholds, test_fpr, test_tpr)))
```

```
====================================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 0.5467330314724699 for threshold 0.807
[[ 3799  1369]
 [ 7260 21072]]
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.3709774265595598 for threshold 0.82
[[1548  998]
 [5440 8514]]
```

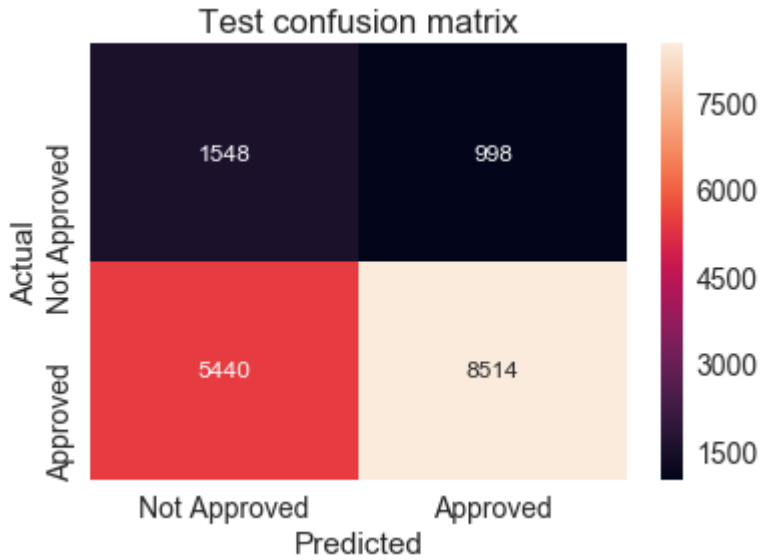
In [246]:
```
import seaborn as snTr
import seaborn as snTe
import pandas as pdH
import matplotlib.pyplot as pltTr
import matplotlib.pyplot as pltTe

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTr=confusion_matrix(y_train, predict(y_train_bow_XB_pred, tr_thresholds, train_fpr, train_tpr))
df_cmTr = pdH.DataFrame(arrayTr,range(2),range(2))
#print(arrayTr)
# https://stackoverflow.com/questions/32723798/how-do-i-add-a-title-to-seaborn-heatmap
axTr = pltTr.axes()

snTr.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html

snTr.heatmap(df_cmTr, annot=True,annot_kws={"size": 12},fmt="d",ax=axTr)# font size, format in digit

labels=['Not Approved','Approved']
axTr.set_xticklabels(labels)
axTr.set_yticklabels(labels)
#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
pltTr.title("Train confusion matrix")
pltTr.xlabel("Predicted")
pltTr.ylabel("Actual")
pltTr.show()

# https://stackoverflow.com/questions/50947776/plot-two-seaborn-heatmap-graphs-side-by-side
```

```
#fig, ax =plt.subplots(1,1)

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTe=confusion_matrix(y_test, predict(y_test_bow_XB_pred, te_thresholds, test_fpr, test_tpr))
df_cmTe = pdH.DataFrame(arrayTe,range(2),range(2))

axTe = pltTe.axes()

snTe.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html
snTe.heatmap(df_cmTe, annot=True,annot_kws={"size": 12},fmt="d",ax=axTe)# font size, format in digit

#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
axTe.set_xticklabels(labels)
axTe.set_yticklabels(labels)
pltTe.title("Test confusion matrix")
pltTe.xlabel("Predicted")
pltTe.ylabel("Actual")
pltTe.show()
```

```
the maximum value of tpr*(1-fpr) 0.5467330314724699 for threshold 0.807
```




```
the maximum value of tpr*(1-fpr) 0.3709774265595598 for threshold 0.82
```



2.5.2 Applying XGBOOST on TFIDF, SET 2

In [0]:
```
# Please write all the code with proper documentation
```

1. Set 2: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(TFIDF)+ preprocessed_eassay (TFIDF)

In [247]:
```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
X_tr_tfidf_XG = hstack((X_train_text_tfidf, X_train_title_tfidf, X_train_schoo
l, X_train_category, X_train_subcategory, X_train_grade, X_train_tea_pfx, X_tr
ain_prj_res_sum, X_train_quantity_norm, X_train_TprevPrj_norm, X_train_price_n
orm)).tocsr()
X_te_tfidf_XG = hstack((X_test_text_tfidf, X_test_title_tfidf , X_test_school,
X_test_category, X_test_subcategory, X_test_grade, X_test_tea_pfx, X_test_prj_
res_sum, X_test_quantity_norm, X_test_TprevPrj_norm, X_test_price_norm)).tocsr
()

print("Final Data matrix | tfidf")
print(X_tr_tfidf_XG.shape, y_train.shape)
print(X_te_tfidf_XG.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix | tfidf
(33500, 8420) (33500,)
(16500, 8420) (16500,)
====================================================================================
======================
```

In [248]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/m
achine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
import xgboost as xgb


xgb_model = xgb.XGBClassifier()

param_grid = {'max_depth': [5,7,15,25],
              'silent': [1],
              'n_estimators': [5,7,15,25], #number of trees, change it to 1000
for better results
             }

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.Gr
idSearchCV.html
modelTfidfXB = GridSearchCV(xgb_model, param_grid, scoring = 'roc_auc', cv=5)
modelTfidfXB.fit(X_tr_tfidf_XG, y_train)

print(modelTfidfXB.best_estimator_)
print(modelTfidfXB.score(X_te_tfidf_XG, y_test))
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.64686120516508
```

In [252]:
```
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html
#print(modelBow.cv_results_)
max_scorestfidfXB=pd.DataFrame(modelTfidfXB.cv_results_).groupby(['param_n_estimators','param_max_depth']).max().unstack()[['mean_test_score','mean_train_score']]
#print(max_scoresBow.mean_train_score)
#print(max_scoresBow.mean_test_score)

print(type(max_scorestfidfXB.mean_test_score))
#print("max_scoresBow",max_scoresBow)
#print(type(max_scoresBow))
#print(max_scoresBow.shape)

print("max_scorestfidfXB.mean_test_score.shape")
print(max_scorestfidfXB.mean_test_score)
print(max_scorestfidfXB.mean_test_score.shape)
print(len(max_scorestfidfXB.mean_test_score))
```

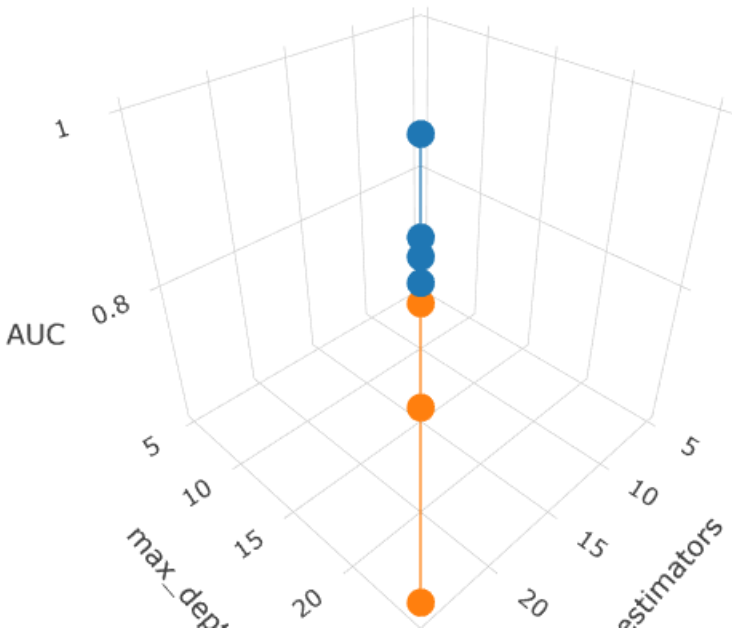
```
<class 'pandas.core.frame.DataFrame'>
max_scorestfidfXB.mean_test_score.shape
param_max_depth           5         7         15        25
param_n_estimators
5                   0.621276  0.621693  0.605339  0.590473
7                   0.626443  0.624549  0.608928  0.597233
15                  0.642220  0.645548  0.632287  0.621011
25                  0.659003  0.662711  0.657335  0.650270
(4, 4)
4
```

In [286]:
```
List_tr_tfidfXB=[]
dfTr=max_scorestfidfXB.mean_train_score
print(len(dfTr))
for i in range(0, len(dfTr)):
    for j in range(0, len(dfTr)):
        if i==j:
            List_tr_tfidfXB.append(dfTr.iloc[i,j])
print(List_tr_tfidfXB)
len(List_tr_tfidfXB)

List_te_tfidfXB=[]
dfTe=max_scorestfidfXB.mean_test_score
print(len(dfTe))
for i in range(0, len(dfTe)):
    for j in range(0, len(dfTe)):
        if i==j:
            List_te_tfidfXB.append(dfTe.iloc[i,j])
print(List_te_tfidfXB)
len(List_te_tfidfXB)
```

```
4
[0.6664244868145422, 0.7321591172896909, 0.979883537082728, 0.999995979422060
1]
4
[0.6212761241900842, 0.624549203675593, 0.6322872785498147, 0.65026957617832]
```

Out[286]: 4

In [287]:
```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
import numpy as np

# https://plot.ly/python/3d-axes/
trace1 = go.Scatter3d(x=[5,7,15,25],y=[5,7,15,25],z=List_tr_tfidfXB, name = 't
rain')
trace2 = go.Scatter3d(x=[5,7,15,25],y=[5,7,15,25],z=List_te_tfidfXB, name = 'C
ross validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```



Conclusion

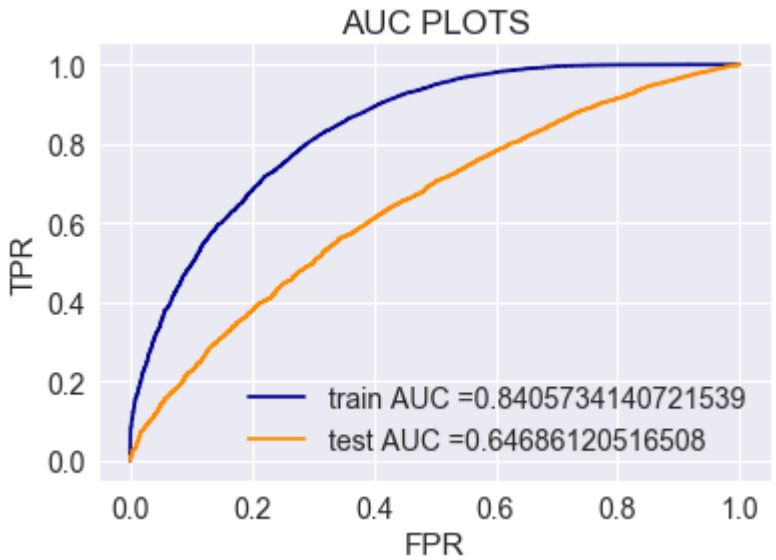
For all the various values of n_estimators=25 and max_depth=7 is giving the best score for test data.

In [255]:
```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
import xgboost as xgb


xgb_model = xgb.XGBClassifier()

param_grid = {'max_depth': [7],
              'silent': [1],
              'n_estimators': [25], #number of trees, change it to 1000 for be
tter results
             }

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.Gr
idSearchCV.html
modelTfidfXBb = GridSearchCV(xgb_model, param_grid, scoring = 'roc_auc', cv=5)
modelTfidfXBb.fit(X_tr_tfidf_XG, y_train)

print(modelTfidfXBb.best_estimator_)
print(modelTfidfXBb.score(X_te_tfidf_XG, y_test))
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.64686120516508
```

In [256]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc

y_train_tfidf_XB_pred = modelTfidfXBb.predict_proba(X_tr_tfidf_XG)[:,1]
y_test_tfidf_XB_pred = modelTfidfXBb.predict_proba(X_te_tfidf_XG)[:,1]

print(modelTfidfXBb.best_estimator_)
print(modelTfidfXBb.score(X_te_tfidf_XG, y_test))

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_tfidf_XB_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_tfidf_XB_pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)),color='darkblue')
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)),color='darkorange')
plt.legend()
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("AUC PLOTS")
plt.grid(True)
plt.show()
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.64686120516508
```



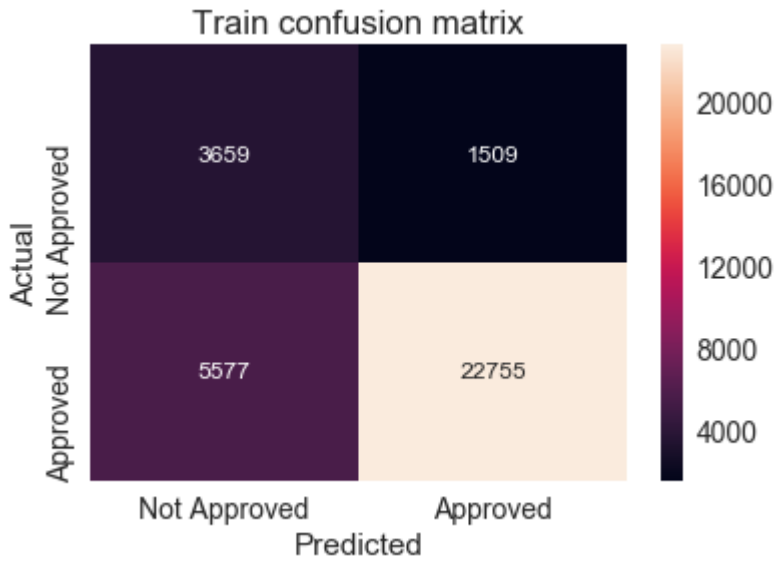
In [257]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
print("="*100)
from sklearn.metrics import confusion_matrix
print("Train confusion matrix")
print(confusion_matrix(y_train, predict(y_train_tfidf_XB_pred, tr_thresholds, train_fpr, train_tpr)))
print("Test confusion matrix")
print(confusion_matrix(y_test, predict(y_test_tfidf_XB_pred, te_thresholds, test_fpr, test_tpr)))
```

```
====================================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 0.56864275628997 for threshold 0.801
[[ 3659  1509]
 [ 5577 22755]]
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.3688939621048668 for threshold 0.829
[[1671  875]
 [6111 7843]]
```

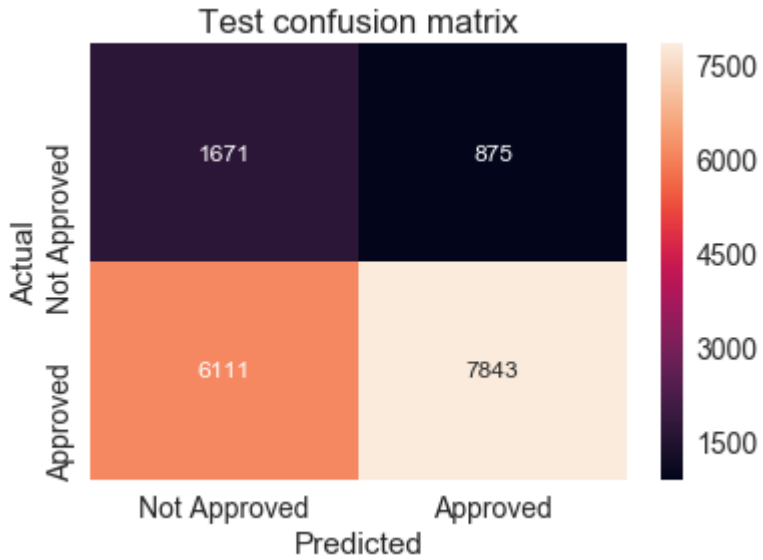
In [258]:
```
import seaborn as snTr
import seaborn as snTe
import pandas as pdH
import matplotlib.pyplot as pltTr
import matplotlib.pyplot as pltTe

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTr=confusion_matrix(y_train, predict(y_train_tfidf_XB_pred, tr_thresholds, train_fpr, train_tpr))
df_cmTr = pdH.DataFrame(arrayTr,range(2),range(2))
#print(arrayTr)
# https://stackoverflow.com/questions/32723798/how-do-i-add-a-title-to-seaborn-heatmap
axTr = pltTr.axes()

snTr.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html

snTr.heatmap(df_cmTr, annot=True,annot_kws={"size": 12},fmt="d",ax=axTr)# font size, format in digit

labels=['Not Approved','Approved']
axTr.set_xticklabels(labels)
axTr.set_yticklabels(labels)
#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
pltTr.title("Train confusion matrix")
pltTr.xlabel("Predicted")
pltTr.ylabel("Actual")
pltTr.show()

# https://stackoverflow.com/questions/50947776/plot-two-seaborn-heatmap-graphs-side-by-side
```

```
#fig, ax =plt.subplots(1,1)

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTe=confusion_matrix(y_test, predict(y_test_tfidf_XB_pred, te_thresholds, test_fpr, test_tpr))
df_cmTe = pdH.DataFrame(arrayTe,range(2),range(2))

axTe = pltTe.axes()

snTe.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html
snTe.heatmap(df_cmTe, annot=True,annot_kws={"size": 12},fmt="d",ax=axTe)# font size, format in digit

#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
axTe.set_xticklabels(labels)
axTe.set_yticklabels(labels)
pltTe.title("Test confusion matrix")
pltTe.xlabel("Predicted")
pltTe.ylabel("Actual")
pltTe.show()
```

```
the maximum value of tpr*(1-fpr) 0.56864275628997 for threshold 0.801
```




```
the maximum value of tpr*(1-fpr) 0.3688939621048668 for threshold 0.829
```



2.5.3 Applying XGBOOST on AVG W2V, SET 3

In [0]:
```
# Please write all the code with proper documentation
```

1. Set 3: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(AVG W2V)+ preprocessed_eassay (AVG W2V)

In [249]:
```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
X_tr_avgW2V_XG = hstack((X_train_essay_avg_w2v, X_train_title_avg_w2v, X_train
_school, X_train_category, X_train_subcategory, X_train_grade, X_train_tea_pfx
, X_train_prj_res_sum, X_train_quantity_norm, X_train_TprevPrj_norm, X_train_p
rice_norm)).tocsr()
X_te_avgW2V_XG = hstack((X_test_essay_avg_w2v, X_test_title_avg_w2v , X_test_s
chool, X_test_category, X_test_subcategory, X_test_grade, X_test_tea_pfx, X_te
st_prj_res_sum, X_test_quantity_norm, X_test_TprevPrj_norm, X_test_price_norm
)).tocsr()

print("Final Data matrix | Avg W2V")
print(X_tr_avgW2V_XG.shape, y_train.shape)
print(X_te_avgW2V_XG.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix | Avg W2V
(33500, 615) (33500,)
(16500, 615) (16500,)
====================================================================================
======================
```

In [250]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/m
achine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
import xgboost as xgb


xgb_model = xgb.XGBClassifier()

param_grid = {'max_depth': [5,7,15,25],
              'silent': [1],
              'n_estimators': [5,7,15,25], #number of trees, change it to 1000
for better results
             }

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.Gr
idSearchCV.html
modelavgw2vXB = GridSearchCV(xgb_model, param_grid, scoring = 'roc_auc', cv=5)
modelavgw2vXB.fit(X_tr_avgW2V_XG, y_train)

print(modelavgw2vXB.best_estimator_)
print(modelavgw2vXB.score(X_te_avgW2V_XG, y_test))
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.6342419025546964
```

In [259]:
```
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html
#print(modelBow.cv_results_)
max_scoresavgw2vXB=pd.DataFrame(modelavgw2vXB.cv_results_).groupby(['param_n_estimators','param_max_depth']).max().unstack()[['mean_test_score','mean_train_score']]
#print(max_scoresBow.mean_train_score)
#print(max_scoresBow.mean_test_score)

print(type(max_scoresavgw2vXB.mean_test_score))
#print("max_scoresBow",max_scoresBow)
#print(type(max_scoresBow))
#print(max_scoresBow.shape)

print("max_scoresavgw2vXB.mean_test_score.shape")
print(max_scoresavgw2vXB.mean_test_score)
print(max_scoresavgw2vXB.mean_test_score.shape)
print(len(max_scoresavgw2vXB.mean_test_score))
```

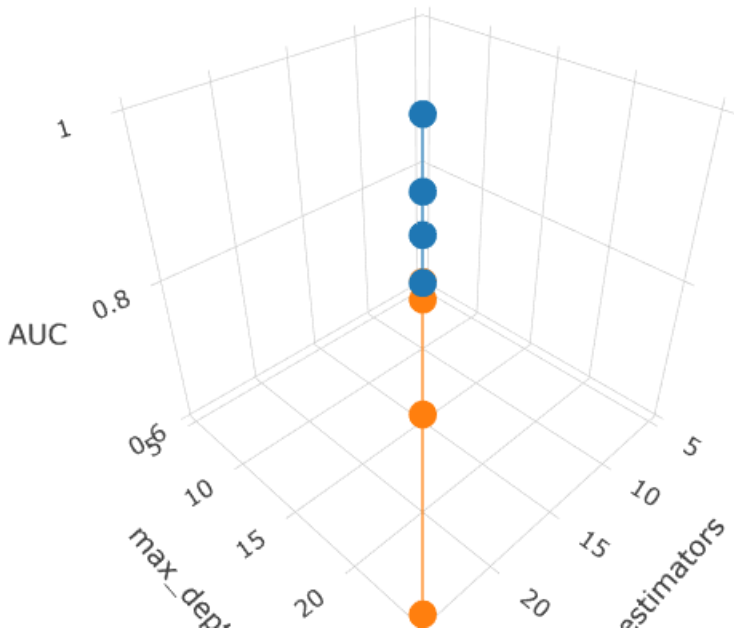
```
<class 'pandas.core.frame.DataFrame'>
max_scoresavgw2vXB.mean_test_score.shape
param_max_depth           5         7         15        25
param_n_estimators
5                  0.610295  0.612606  0.581315  0.575728
7                  0.618133  0.618933  0.586409  0.585549
15                 0.633126  0.632455  0.608625  0.604740
25                 0.645870  0.647070  0.624482  0.620107
(4, 4)
4
```

In [288]:
```
List_tr_avgw2vXB=[]
dfTr=max_scoresavgw2vXB.mean_train_score
print(len(dfTr))
for i in range(0, len(dfTr)):
    for j in range(0, len(dfTr)):
        if i==j:
            List_tr_avgw2vXB.append(dfTr.iloc[i,j])
print(List_tr_avgw2vXB)
len(List_tr_avgw2vXB)

List_te_avgw2vXB=[]
dfTe=max_scoresavgw2vXB.mean_test_score
print(len(dfTe))
for i in range(0, len(dfTe)):
    for j in range(0, len(dfTe)):
        if i==j:
            List_te_avgw2vXB.append(dfTe.iloc[i,j])
print(List_te_avgw2vXB)
len(List_te_avgw2vXB)
```

```
4
[0.6923653572732837, 0.7934984584300324, 0.9996303528074207, 0.99999993170191
34]
4
[0.6102948529119696, 0.618933452513227, 0.6086250907322269, 0.620107288556440
3]
```

Out[288]: 4

In [289]:
```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
import numpy as np

# https://plot.ly/python/3d-axes/
trace1 = go.Scatter3d(x=[5,7,15,25],y=[5,7,15,25],z=List_tr_avgw2vXB, name =
'train')
trace2 = go.Scatter3d(x=[5,7,15,25],y=[5,7,15,25],z=List_te_avgw2vXB, name =
'Cross validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```



Conclusion

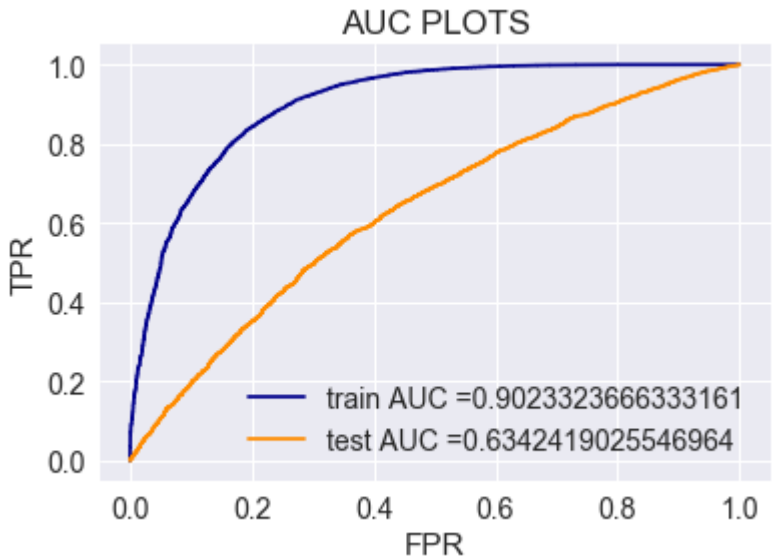
For all the various values of n_estimators=25 and max_depth=7 is giving the best score for test data.

In [262]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
import xgboost as xgb


xgb_model = xgb.XGBClassifier()

param_grid = {'max_depth': [7],
              'silent': [1],
              'n_estimators': [25], #number of trees, change it to 1000 for better results
             }

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modelavgw2vXBb = GridSearchCV(xgb_model, param_grid, scoring = 'roc_auc', cv=5)
modelavgw2vXBb.fit(X_tr_avgW2V_XG, y_train)

print(modelavgw2vXBb.best_estimator_)
print(modelavgw2vXBb.score(X_te_avgW2V_XG, y_test))
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.6342419025546964
```

In [263]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc

y_train_tfidf_XB_pred = modelavgw2vXBb.predict_proba(X_tr_avgW2V_XG)[:,1]
y_test_tfidf_XB_pred = modelavgw2vXBb.predict_proba(X_te_avgW2V_XG)[:,1]

print(modelavgw2vXBb.best_estimator_)
print(modelavgw2vXBb.score(X_te_avgW2V_XG, y_test))

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_tfidf_XB_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_tfidf_XB_pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)),color='darkblue')
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)),color='darkorange')
plt.legend()
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("AUC PLOTS")
plt.grid(True)
plt.show()
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.6342419025546964
```



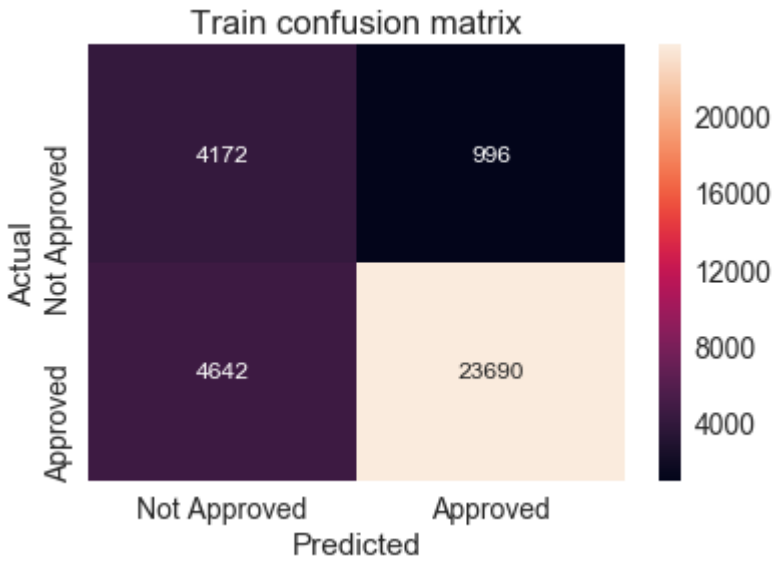
In [264]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
print("="*100)
from sklearn.metrics import confusion_matrix
print("Train confusion matrix")
print(confusion_matrix(y_train, predict(y_train_tfidf_XB_pred, tr_thresholds, train_fpr, train_tpr)))
print("Test confusion matrix")
print(confusion_matrix(y_test, predict(y_test_tfidf_XB_pred, te_thresholds, test_fpr, test_tpr)))
```

```
====================================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 0.6750090916680545 for threshold 0.803
[[ 4172   996]
 [ 4642 23690]]
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.36476275262418173 for threshold 0.83
[[1611  935]
 [5910 8044]]
```

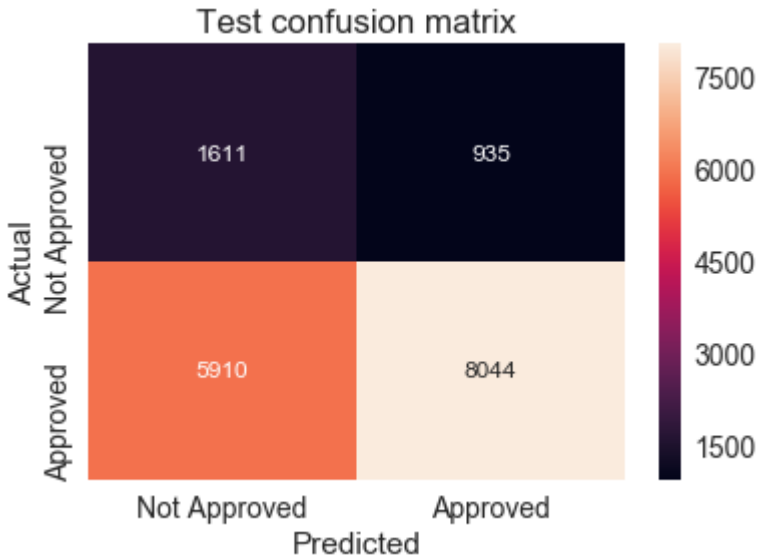
In [265]:
```
import seaborn as snTr
import seaborn as snTe
import pandas as pdH
import matplotlib.pyplot as pltTr
import matplotlib.pyplot as pltTe

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTr=confusion_matrix(y_train, predict(y_train_tfidf_XB_pred, tr_thresholds, train_fpr, train_tpr))
df_cmTr = pdH.DataFrame(arrayTr,range(2),range(2))
#print(arrayTr)
# https://stackoverflow.com/questions/32723798/how-do-i-add-a-title-to-seaborn-heatmap
axTr = pltTr.axes()

snTr.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html

snTr.heatmap(df_cmTr, annot=True,annot_kws={"size": 12},fmt="d",ax=axTr)# font size, format in digit

labels=['Not Approved','Approved']
axTr.set_xticklabels(labels)
axTr.set_yticklabels(labels)
#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
pltTr.title("Train confusion matrix")
pltTr.xlabel("Predicted")
pltTr.ylabel("Actual")
pltTr.show()

# https://stackoverflow.com/questions/50947776/plot-two-seaborn-heatmap-graphs-side-by-side
```

```
#fig, ax =plt.subplots(1,1)

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTe=confusion_matrix(y_test, predict(y_test_tfidf_XB_pred, te_thresholds, test_fpr, test_tpr))
df_cmTe = pdH.DataFrame(arrayTe,range(2),range(2))

axTe = pltTe.axes()

snTe.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html
snTe.heatmap(df_cmTe, annot=True,annot_kws={"size": 12},fmt="d",ax=axTe)# font size, format in digit

#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
axTe.set_xticklabels(labels)
axTe.set_yticklabels(labels)
pltTe.title("Test confusion matrix")
pltTe.xlabel("Predicted")
pltTe.ylabel("Actual")
pltTe.show()
```

```
the maximum value of tpr*(1-fpr) 0.6750090916680545 for threshold 0.803
```




```
the maximum value of tpr*(1-fpr) 0.36476275262418173 for threshold 0.83
```



2.5.4 Applying XGBOOST on TFIDF W2V, SET 4

In [0]:
```
# Please write all the code with proper documentation
```

1. Set 4: categorical(instead of one hot encoding, try response coding (https://www.appliedaicourse.com/course/applied-ai-course-online/lessons/handling-categorical-and-numerical-features/): use probability values), numerical features + project_title(TFIDF W2V)+ preprocessed_eassay (TFIDF W2V)

In [266]:
```
# merge two sparse matrices: https://stackoverflow.com/a/19710648/4084039
from scipy.sparse import hstack
X_tr_tfidf_W2V_XG = hstack((tr_tfidf_w2v_essay_vectors, tr_tfidf_w2v_title_vectors, X_train_school, X_train_category, X_train_subcategory, X_train_grade, X_train_tea_pfx, X_train_prj_res_sum, X_train_quantity_norm, X_train_TprevPrj_norm, X_train_price_norm)).tocsr()
X_te_tfidf_W2V_XG = hstack((te_tfidf_w2v_essay_vectors, te_tfidf_w2v_title_vectors , X_test_school, X_test_category, X_test_subcategory, X_test_grade, X_test_tea_pfx, X_test_prj_res_sum, X_test_quantity_norm, X_test_TprevPrj_norm, X_test_price_norm)).tocsr()

print("Final Data matrix | TFIDF W2V")
print(X_tr_tfidf_W2V_XG.shape, y_train.shape)
print(X_te_tfidf_W2V_XG.shape, y_test.shape)
print("="*100)
```

```
Final Data matrix | TFIDF W2V
(33500, 615) (33500,)
(16500, 615) (16500,)
====================================================================================================
```

In [267]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
import xgboost as xgb


xgb_model = xgb.XGBClassifier()

param_grid = {'max_depth': [5,7,15,25],
              'silent': [1],
              'n_estimators': [5,7,15,25], #number of trees, change it to 1000 for better results
             }

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modeltfidf_w2vXB = GridSearchCV(xgb_model, param_grid, scoring = 'roc_auc', cv =5)
modeltfidf_w2vXB.fit(X_tr_tfidf_W2V, y_train)

print(modeltfidf_w2vXB.best_estimator_)
print(modeltfidf_w2vXB.score(X_te_tfidf_W2V, y_test))
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=5, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.6371488279129687
```

here

In [268]:
```
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.unstack.html
#print(modelBow.cv_results_)
max_scorestfidf_w2vXB=pd.DataFrame(modeltfidf_w2vXB.cv_results_).groupby(['param_n_estimators','param_max_depth']).max().unstack()[['mean_test_score','mean_train_score']]
#print(max_scoresBow.mean_train_score)
#print(max_scoresBow.mean_test_score)

print(type(max_scorestfidf_w2vXB.mean_test_score))
#print("max_scoresBow",max_scoresBow)
#print(type(max_scoresBow))
#print(max_scoresBow.shape)

print("max_scorestfidf_w2vXB.mean_test_score.shape")
print(max_scorestfidf_w2vXB.mean_test_score)
print(max_scorestfidf_w2vXB.mean_test_score.shape)
print(len(max_scorestfidf_w2vXB.mean_test_score))
```

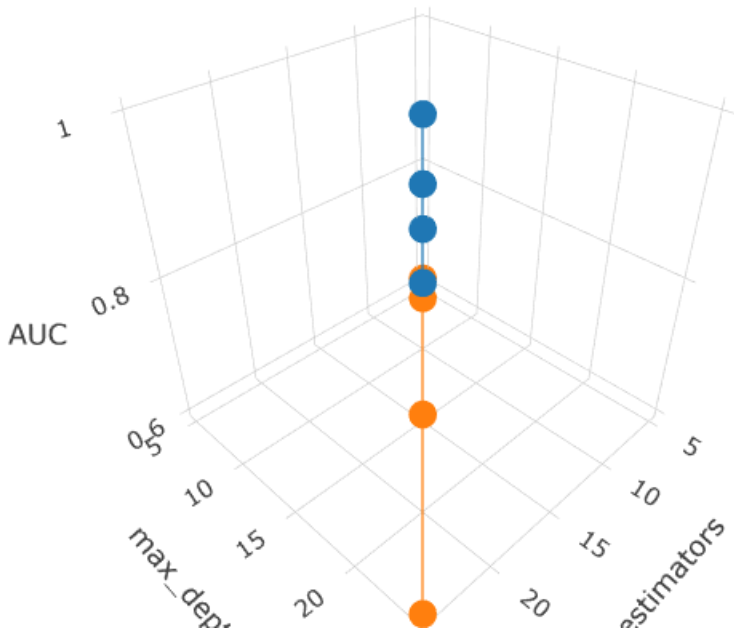
```
<class 'pandas.core.frame.DataFrame'>
max_scorestfidf_w2vXB.mean_test_score.shape
param_max_depth           5         7         15        25
param_n_estimators
5                   0.608111  0.610122  0.581387  0.568759
7                   0.615186  0.612713  0.586287  0.580566
15                  0.628689  0.627898  0.599888  0.599863
25                  0.641858  0.640739  0.615216  0.612230
(4, 4)
4
```

In [290]:
```
List_tr_tfidf_w2vXB =[]
dfTr=max_scorestfidf_w2vXB.mean_train_score
print(len(dfTr))
for i in range(0, len(dfTr)):
    for j in range(0, len(dfTr)):
        if i==j:
            List_tr_tfidf_w2vXB .append(dfTr.iloc[i,j])
print(List_tr_tfidf_w2vXB )
len(List_tr_tfidf_w2vXB )

List_te_tfidf_w2vXB =[]
dfTe=max_scorestfidf_w2vXB.mean_test_score
print(len(dfTe))
for i in range(0, len(dfTe)):
    for j in range(0, len(dfTe)):
        if i==j:
            List_te_tfidf_w2vXB .append(dfTe.iloc[i,j])
print(List_te_tfidf_w2vXB )
len(List_te_tfidf_w2vXB )
```

```
4
[0.695916201948215, 0.8003467752170167, 0.9995440317639874, 0.999999953041113
8]
4
[0.6081105653395634, 0.6127128767963513, 0.5998875263872466, 0.6122302407513
6]
```

Out[290]: 4

In [291]:
```
import plotly.offline as offline
import plotly.graph_objs as go
offline.init_notebook_mode()
import numpy as np

# https://plot.ly/python/3d-axes/
trace1 = go.Scatter3d(x=[5,7,15,25],y=[5,7,15,25],z=List_tr_tfidf_w2vXB , name
= 'train')
trace2 = go.Scatter3d(x=[5,7,15,25],y=[5,7,15,25],z=List_te_tfidf_w2vXB , name
= 'Cross validation')
data = [trace1, trace2]

layout = go.Layout(scene = dict(
        xaxis = dict(title='n_estimators'),
        yaxis = dict(title='max_depth'),
        zaxis = dict(title='AUC'),))

fig = go.Figure(data=data, layout=layout)
offline.iplot(fig, filename='3d-scatter-colorscale')
```



Conclusion

For all the various values of n_estimators=25 and max_depth=7 is giving the best score for test data.

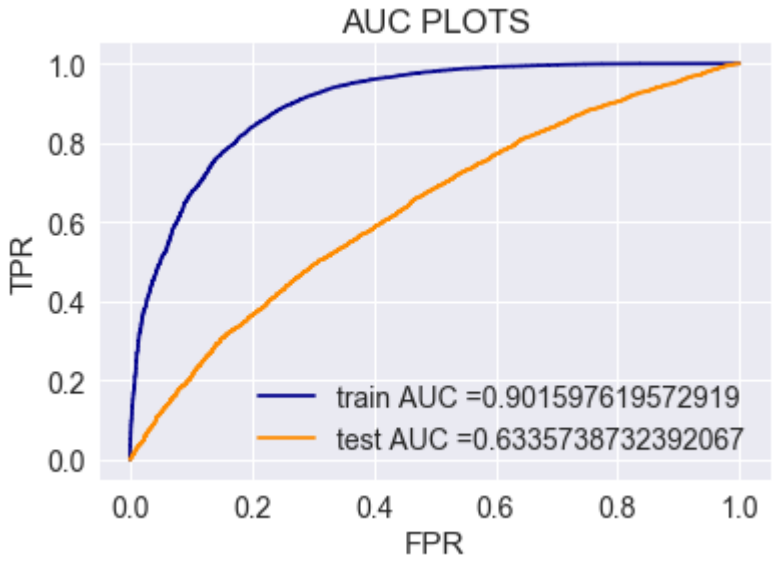
In [271]:
```
#code source: http://occam.olin.edu/sites/default/files/DataScienceMaterials/machine_learning_lecture_2/Machine%20Learning%20Lecture%202.html
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import *
import xgboost as xgb


xgb_model = xgb.XGBClassifier()

param_grid = {'max_depth': [7],
              'silent': [1],
              'n_estimators': [25], #number of trees, change it to 1000 for better results
             }

#Using GridSearchCV
# https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
modeltfidf_w2vXBb = GridSearchCV(xgb_model, param_grid, scoring = 'roc_auc', cv=5)
modeltfidf_w2vXBb.fit(X_tr_tfidf_W2V, y_train)

print(modeltfidf_w2vXBb.best_estimator_)
print(modeltfidf_w2vXBb.score(X_te_tfidf_W2V, y_test))
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.6335738732392067
```

to here

In [272]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve
from sklearn.metrics import roc_curve, auc

y_train_tfidf_XB_pred = modeltfidf_w2vXBb.predict_proba(X_tr_tfidf_W2V)[:,1]
y_test_tfidf_XB_pred = modeltfidf_w2vXBb.predict_proba(X_te_tfidf_W2V)[:,1]

print(modeltfidf_w2vXBb.best_estimator_)
print(modeltfidf_w2vXBb.score(X_te_tfidf_W2V, y_test))

train_fpr, train_tpr, tr_thresholds = roc_curve(y_train, y_train_tfidf_XB_pred)
test_fpr, test_tpr, te_thresholds = roc_curve(y_test, y_test_tfidf_XB_pred)

plt.plot(train_fpr, train_tpr, label="train AUC ="+str(auc(train_fpr, train_tpr)),color='darkblue')
plt.plot(test_fpr, test_tpr, label="test AUC ="+str(auc(test_fpr, test_tpr)),color='darkorange')
plt.legend()
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("AUC PLOTS")
plt.grid(True)
plt.show()
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=7, min_child_weight=1, missing=None,
       n_estimators=25, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=1, subsample=1,
       verbosity=1)
0.6335738732392067
```



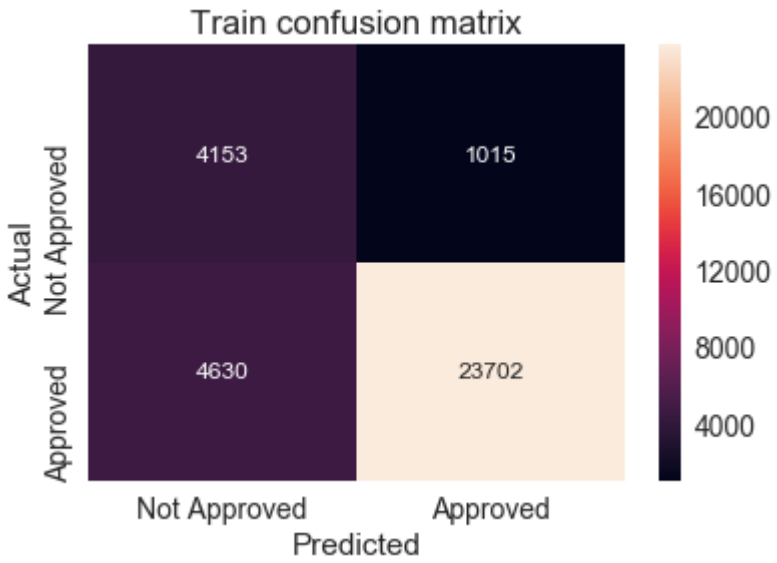
In [273]:
```
# https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
print("="*100)
from sklearn.metrics import confusion_matrix
print("Train confusion matrix")
print(confusion_matrix(y_train, predict(y_train_tfidf_XB_pred, tr_thresholds, train_fpr, train_tpr)))
print("Test confusion matrix")
print(confusion_matrix(y_test, predict(y_test_tfidf_XB_pred, te_thresholds, test_fpr, test_tpr)))
```

```
====================================================================================================
Train confusion matrix
the maximum value of tpr*(1-fpr) 0.6722753489255441 for threshold 0.797
[[ 4153  1015]
 [ 4630 23702]]
Test confusion matrix
the maximum value of tpr*(1-fpr) 0.35280870678103937 for threshold 0.827
[[1583  963]
 [6036 7918]]
```

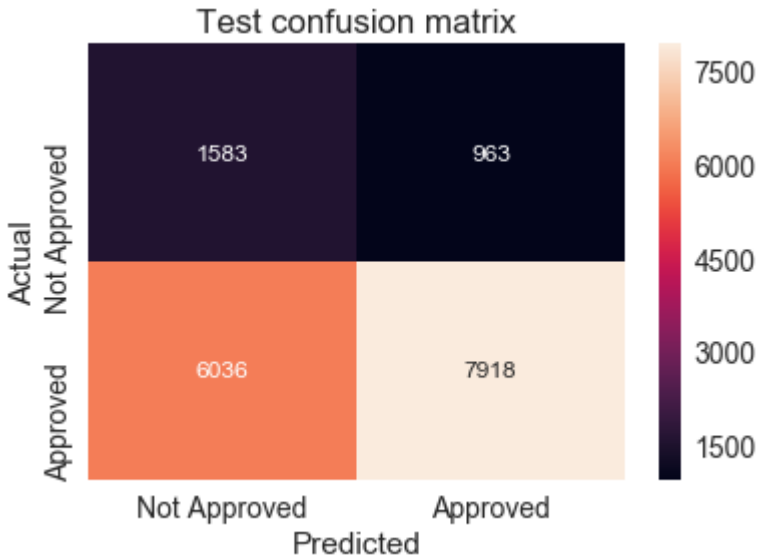
In [274]:
```
import seaborn as snTr
import seaborn as snTe
import pandas as pdH
import matplotlib.pyplot as pltTr
import matplotlib.pyplot as pltTe

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matrix
arrayTr=confusion_matrix(y_train, predict(y_train_tfidf_XB_pred, tr_thresholds, train_fpr, train_tpr))
df_cmTr = pdH.DataFrame(arrayTr,range(2),range(2))
#print(arrayTr)
# https://stackoverflow.com/questions/32723798/how-do-i-add-a-title-to-seaborn-heatmap
axTr = pltTr.axes()

snTr.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html

snTr.heatmap(df_cmTr, annot=True,annot_kws={"size": 12},fmt="d",ax=axTr)# font size, format in digit

labels=['Not Approved','Approved']
axTr.set_xticklabels(labels)
axTr.set_yticklabels(labels)
#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
pltTr.title("Train confusion matrix")
pltTr.xlabel("Predicted")
pltTr.ylabel("Actual")
pltTr.show()

# https://stackoverflow.com/questions/50947776/plot-two-seaborn-heatmap-graphs
```

```
-side-by-side
#fig, ax =plt.subplots(1,1)

# https://stackoverflow.com/questions/35572000/how-can-i-plot-a-confusion-matr
ix
arrayTe=confusion_matrix(y_test, predict(y_test_tfidf_XB_pred, te_thresholds,
test_fpr, test_tpr))
df_cmTe = pdH.DataFrame(arrayTe,range(2),range(2))

axTe = pltTe.axes()

snTe.set(font_scale=1.4)#for label size

# https://seaborn.pydata.org/generated/seaborn.heatmap.html
snTe.heatmap(df_cmTe, annot=True,annot_kws={"size": 12},fmt="d",ax=axTe)# font
size, format in digit

#Suggestion 4.Label confusion matrix heatmap with actual and predicted labels.
axTe.set_xticklabels(labels)
axTe.set_yticklabels(labels)
pltTe.title("Test confusion matrix")
pltTe.xlabel("Predicted")
pltTe.ylabel("Actual")
pltTe.show()
```

```
the maximum value of tpr*(1-fpr) 0.6722753489255441 for threshold 0.797
```




```
the maximum value of tpr*(1-fpr) 0.35280870678103937 for threshold 0.827
```



3. Conclusion

In [275]:
```
# Please compare all your models using Prettytable library
from prettytable import PrettyTable

x = PrettyTable()

x.field_names = ["Vectorizer", "Algorithm", "max_depth", "n_estimator", "Test
 AUC"]

x.add_row(["BOW", "Random Forest", 500, 500, 0.680534 ])
x.add_row(["TFIDF", "Random Forest", 100, 100, 0.645548 ])
x.add_row(["AVG W2V)", "Random Forest", 50, 500, 0.6344669 ])
x.add_row(["TFIDF W2V", "Random Forest", 50, 500, 0.61887161 ])
x.add_row(["BOW", "XGBoost", 7, 25, 0.6463364 ])
x.add_row(["TFIDF", "XGBoost", 7, 25, 0.6468612 ])
x.add_row(["AVG W2V)", "XGBoost", 7, 25, 0.63424190 ])
x.add_row(["TFIDF W2V", "XGBoost", 7, 25, 0.63357387 ])

print(x)
```

```
+------------+---------------+-----------+-------------+------------+
| Vectorizer |   Algorithm   | max_depth | n_estimator |  Test AUC  |
+------------+---------------+-----------+-------------+------------+
|    BOW     | Random Forest |    500    |     500     |  0.680534  |
|   TFIDF    | Random Forest |    100    |     100     |  0.645548  |
|  AVG W2V)  | Random Forest |     50    |     500     | 0.6344669  |
| TFIDF W2V  | Random Forest |     50    |     500     | 0.61887161 |
|    BOW     |    XGBoost    |     7     |      25     | 0.6463364  |
|   TFIDF    |    XGBoost    |     7     |      25     | 0.6468612  |
|  AVG W2V)  |    XGBoost    |     7     |      25     | 0.6342419  |
| TFIDF W2V  |    XGBoost    |     7     |      25     | 0.63357387 |
+------------+---------------+-----------+-------------+------------+
```

4.Summary

Step followed

- Preprocessing of Project_subject_categories
- Preprocessing of Project_subject_subcategories
- Preprocessing of Project_grade_category
- Preprocessing of teacher_prefix
- Text Preprocessing for Project essay and Project Title

- Took first 50000 data points for doing the assignment and removed the Class lable (Project_is_approved)
- Took data points for doing the assignment and separate the Class lable (Project_is_approved)
- Splitting Data into Train (further split into Train and Cross validation) and Test.
- Making datamodel ready ##### categorical features

```
          - For School_State do Response Encoding
            a) Take the exact count for all unique School_State, per project
is approved
            b) Take the exact count for all unique School_State, per project
is not approved
            b) Take the exact count for all unique School_State, irrespectiv
e of project is approved or not
            c) Create a response Table, such that, for a given Value for Pro
ject is approved(a)/Total Number project                              approve
 or not (c)
            d) Create a response Table, such that, for a given Value for Pro
ject is not approved(b)/Total Number project                          approve
 or not (c)
            e) basically, with each school_state, we are finding the probabi
lity against project approved or not.
            f) delete all the unwanted cell (nan) from both Train data
            g) Create a new dataframe from Actual Train and Test data, which
has only School State and Project_is_approved                        column
            h) Above created Train Dataframe, merge it with Train_response d
ataframe, created in Step a to Step f
            i) Above created Test Dataframe, merge it with Train_response da
taframe, created in Step a to Step f
            j) Replace the unwanted coulmn (NAN) with 0.5 value for both Tra
in and Test dataframe, which we created above
            k) Roundoff with 2 decimal points

          - Do the above a to k steps for Category also.
          - Do the above a to k steps for Sub Category also.
          - Do the above a to k steps for Grade Category also.
          - Do the above a to k steps for Project_resource_Summary also.
```

numeric

```
            - encoding of price is splited into Train and Test vector
            - encoding of teacher_number_of_previously_posted_projects is splited i
nto Train and Test vector
            - encoding of quantity is splited into Train and Test vector
```

Text features

```
            - encoding of project_essay(BOW) is splited into Train and Test vector
            - encoding of project_title(BOW) is splited into Train and Test vector
            - encoding of project_essay(TFIDF) is splited into Train and Test vecto
r
            - encoding of project_title(TFIDF) is splited into Train and Test vecto
r
            - encoding of project_essay(Avg W2V) is splited into Train and Test vec
tor
            - encoding of project_title(Avg W2V) is splited into Train and Test vec
tor
            - encoding of project_essay(TFIDF W2V) is splited into Train and Test v
ector
            - encoding of project_title(TFIDF W2V) is splited into Train and Test v
ector
```

Applying RandomForestClassifier

For SET 1

- Merging all the above features for SET 1 Horizontally merging(with hstack) all categorical (response coding), numerical features + project_title(BOW) + preprocessed_essay (BOW)
- Fit a model on on train (on above merge features) data by using GridSearchCV(RandomForestClassifier(class_weight="balanced"))
- take the mean_train and mean-test value from the above fit model.
- Draw 3-D scatter graph for both train and test data. between max_depth, n_estimator and AUC(mean_test_score).
- Choose best max_depth and n_estimator from best*estimator* function
- Draw roc_auc graph
- Create Confusion matrix, in heatmap

For SET 2

- Merging all the above features for SET 2 Horizontally merging(with hstack) all categorical (response coding), numerical features + project_title(TFIDF) + preprocessed_essay (TFIDF)
- Fit a model on on train (on above merge features) data by using GridSearchCV(RandomForestClassifier(class_weight="balanced"))
- take the mean_train and mean-test value from the above fit model.
- Draw 3-D scatter graph for both train and test data. between max_depth, n_estimator and AUC(mean_test_score).

- Choose best max_depth and n_estimator from best*estimator* function
- Draw roc_auc graph
- Create Confusion matrix, in heatmap

For SET 3

- Merging all the above features for SET 3 Horizontally merging(with hstack) all categorical (response coding), numerical features + project_title(AVG W2V) + preprocessed_essay (AVG W2V)
- Fit a model on on train (on above merge features) data by using GridSearchCV(RandomForestClassifier(class_weight="balanced"))
- take the mean_train and mean-test value from the above fit model.
- Draw 3-D scatter graph for both train and test data. between max_depth, n_estimator and mean_test_score.
- Choose best max_depth and n_estimator from best*estimator* function
- Draw roc_auc graph
- Create Confusion matrix, in heatmap

For SET 4

- Merging all the above features for SET 4 Horizontally merging(with hstack) all categorical (response coding), numerical features + project_title(TFIDF W2V) + preprocessed_essay (TFIDF W2V)
- Fit a model on on train (on above merge features) data by using GridSearchCV(RandomForestClassifier(class_weight="balanced"))
- take the mean_train and mean-test value from the above fit model.
- Draw 3-D scatter graph for both train and test data. between max_depth, n_estimator and mean_test_score.
- Choose best max_depth and n_estimator from best*estimator* function
- Draw roc_auc graph
- Create Confusion matrix, in heatmap

Applying XGBOOST

For SET 1

- Merging all the above features for SET 1 Horizontally merging(with hstack) all categorical (response coding), numerical features + project_title(BOW) + preprocessed_essay (BOW)
- Fit a model on on train (on above merge features) data by using GridSearchCV(XGBClassifier())
- take the mean_train and mean-test value from the above fit model.
- Draw 3-D scatter graph for both train and test data. between max_depth, n_estimator and AUC(mean_test_score).
- Choose best max_depth and n_estimator from best*estimator* function
- Draw roc_auc graph
- Create Confusion matrix, in heatmap

For SET 2

- Merging all the above features for SET 2 Horizontally merging(with hstack) all categorical (response coding), numerical features + project_title(TFIDF) + preprocessed_essay (TFIDF)
- Fit a model on on train (on above merge features) data by using GridSearchCV(XGBClassifier())

- take the mean_train and mean-test value from the above fit model.
- Draw 3-D scatter graph for both train and test data. between max_depth, n_estimator and AUC(mean_test_score).
- Choose best max_depth and n_estimator from best*estimator* function
- Draw roc_auc graph
- Create Confusion matrix, in heatmap

For SET 3

- Merging all the above features for SET 3 Horizontally merging(with hstack) all categorical (response coding), numerical features + project_title(AVG W2V) + preprocessed_essay (AVG W2V)
- Fit a model on on train (on above merge features) data by using GridSearchCV(XGBClassifier())
- take the mean_train and mean-test value from the above fit model.
- Draw 3-D scatter graph for both train and test data. between max_depth, n_estimator and AUC(mean_test_score).
- Choose best max_depth and n_estimator from best*estimator* function
- Draw roc_auc graph
- Create Confusion matrix, in heatmap

For SET 4

- Merging all the above features for SET 4 Horizontally merging(with hstack) all categorical (response coding), numerical features + project_title(TFIDF W2V) + preprocessed_essay (TFIDF W2V)
- Fit a model on on train (on above merge features) data by using GridSearchCV(XGBClassifier())
- take the mean_train and mean-test value from the above fit model.
- Draw 3-D scatter graph for both train and test data. between max_depth, n_estimator and AUC(mean_test_score).
- Choose best max_depth and n_estimator from best*estimator* function
- Draw roc_auc graph
- Create Confusion matrix, in heatmap